Hujun Yin Peter Tino
Emilio Corchado Will Byrne
Xin Yao (Eds.)

Intelligent Data Engineering and Automated Learning – IDEAL 2007

LNCS 4881

8th International Conference
Birmingham, UK, December 2007
Proceedings

Springer

Lecture Notes in Computer Science 4881

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Hujun Yin Peter Tino Emilio Corchado
Will Byrne Xin Yao (Eds.)

Intelligent Data Engineering and Automated Learning – IDEAL 2007

8th International Conference
Birmingham, UK, December 16-19, 2007
Proceedings

Springer

Volume Editors

Hujun Yin
The University of Manchester
Manchester, M60 1QD, UK
E-mail: hujun.yin@manchester.ac.uk

Peter Tino
Will Byrne
Xin Yao
University of Birmingham
Birmingham B15 2TT, UK
E-mail: {p.tino, w.f.byrne, x.yao}@cs.bham.ac.uk

Emilio Corchado
University of Burgos
09001 Burgos, Spain
E-mail: escorchado@ubu.es

Library of Congress Control Number: 2007941157

CR Subject Classification (1998): H.2.8, F.2.2, I.2, F.4, K.4.4, H.3, H.4

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-540-77225-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-77225-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12202126 06/3180 5 4 3 2 1 0

Preface

After a vibrant and successful event in Burgos, Spain, last year, this year's international conference on Intelligent Data Engineering and Automated Learning—IDEAL 2007 (http://events.cs.bham.ac.uk/ideal07/)—was held in the second largest city of the UK, Birmingham. The IDEAL conference has become a unique multidisciplinary forum for researchers in both theoretical and practical aspects of learning and information processing, data mining, retrieval and management, bioinformatics and bio-inspired models, agents and hybrid systems, and financial engineering. A special feature of the IDEAL conferences is the cross-disciplinary exchange of ideas in emerging techniques and applications in these areas. Data engineering and associated learning paradigms are playing increasingly important roles in an increasing number of disciplines and fields. The multidisciplinary nature of contemporary research and modern technology is pushing boundaries and one of the principal aims of the IDEAL conference is to promote interactions and collaborations across disciplines.

This volume of [illegible] contains accepted papers presented at IDEAL 2007 held at the University of Birmingham, UK, during December 16–19, 2007. This year, the conference received over 270 submissions from around the world, which were subsequently peer-refereed by the Programme Committee comprising leading scholars in the field. Each paper was rigorously reviewed by two reviewers and only papers that had received positive comments from both reviewers were accepted and included in the proceedings in order to maintain the highest quality of the conference. This resulted in about 110 top quality papers for the conference and the proceedings. The acceptance rate was about 40%. The buoyant numbers of submissions in recent years are a clear indication of the importance of the fields related to IDEAL and the popularity of the IDEAL conference and community. This year's conference had five regular themes: Learning and Information Processing, Data Mining and Information Management, Bioinformatics and Neuroinformatics, Agents and Distributed Systems, and Financial Engineering and Modelling. Two special sessions, Agent-based Approach to Service Sciences and Neural-evolutionary Fusion Algorithms and Their Application, were also organized by Akira Namatame and Sancho Salcedo-Sanz, respectively. Many new ideas, novel algorithms and emerging techniques were reported and discussed at the conference.

This 8th IDEAL conference also enjoyed outstanding keynote speeches by distinguished guest speakers: Piero Bonissone, GE Global Research, Kevin Kelly, Carnegie Mellon University, Toby Gibson, European Molecular Biology Laboratory, and Soo-Young Lee of the Korea Advanced Institute of Science and Technology. Their in-depth coverage and work on various challenging topics served as both exceptional examples and inspirations for others.

This year IDEAL also teamed up with two international journals, namely, the [illegible] and the [illegible] for two special issues. The extended papers, together with contributed articles received in response to subsequent open calls, will go through further rounds of peer refereeing in the remits of these two journals.

We would like to thank the International Advisory Committee and the Steering Committee for the guidance and advice. We would particularly like to acknowledge the diligent work of our Programme Committee members, who performed review tasks admirably under tight deadline pressures.

Particular thanks go to CERCIA (The Centre of Excellence for Research in Computational Intelligence and Applications) and the School of Computer Science (especially Ceinwen Cushway) of the University of Birmingham, for their support and organization of IDEAL 2007.

We are also grateful to the publisher, Springer, especially Alfred Hofmann and Anna Kramer at the LNCS Editorial Office, for their continued support and collaboration in this demanding publication project.

Last but not the least we thank all the authors and participants for their contributions that made this conference such a successful and enjoyable event.

October 2007

Hujun Yin
Peter Tino
Emilio Corchado
Will Byrne
Xin Yao

Organization

General Co-chairs

Xin Yao	University of Birmingham, UK
Benjamin Wah	University of Illinois, Urbana-Champaign, USA

International Advisory Committee

Lei Xu (Chair)	Chinese University of Hong Kong
Yaser Abu-Mostafa	CALTECH, USA
Shun-ichi Amari	RIKEN, Japan
Michael Dempster	University of Cambridge, UK
Sun-Yung Kung	Princeton University, USA
Erkki Oja	Helsinki University of Technology, Finland
Latit M. Patnaik	Indian Institute of Science, India

Steering Committee

Hujun Yin (Co-chair)	University of Manchester, UK
Lai-Wan Chan (Co-chair)	Chinese University of Hong Kong
Nigel Allinson	University of Sheffield, UK
Yiu-ming Cheung	Hong Kong Baptist University, Hong Kong
Emilio Corchado	University of Burgos, Spain
Marcus Gallagher	University of Queensland, Australia
Marc van Hulle	K.U. Leuven, Belgium
John Keane	University of Manchester, UK
Jimmy Lee	Chinese University of Hong Kong, Hong Kong
Malik Magdon-Ismail	Rensselaer Polytechnic Institute, USA
Zheng Rong Yang	University of Exeter, UK
Ron Sun	Rensselaer Polytechnic Institute, USA
Ning Zhong	Maebashi Institute of Technology, Japan

Programme Committee Chairs

Hujun Yin (Chair)	University of Manchester, UK
Peter Tino (Technical Chair)	University of Birmingham, UK
Emilio Corchado (Technical Co-chair)	University of Burgos, Spain
Malik Magdon-Ismail (Technical Co-chair)	Rensselaer Polytechnic Institute, USA
Zheng Rong Yang (Technical Co-chair)	University of Exeter, UK
Will Byrne (Organizing Chair)	University of Birmingham, UK

Programme Committee

Ajith Abraham
José Adserias
Khurshid Ahmad
Nigel Allinson
Ángel Alonso
Luis Alonso
Martyn Amos
Davide Anguita
Bruno Apolloni
Jiyuan An
Javier Bajo
Federico Barber
Bruno Baruque
Lubica Benuskova
Michael Biehl
Alan Blair
Mikael Boden
Lourdes Borrajo
Juan Botía
Vicente Botti
Teodoro Calonge
Carlos Carrascosa
Andre de Carvalho
Matthew Casey
Sheng Chen
Songcan Chen
Sung-Bae Cho
Sungzoon Cho
Juan Manuel Corchado
Rafael Corchuelo
David Corne
Ernesto Costa
Robert Dale
Bernard De Baets
Yanira De Paz
Ricardo Del Olmo
Miguel Delgado
Fernando Díaz
Zhao Yang Dong
José Dorronsoro
Richard Everson
Igor Farkas
Fernando Fernández
Florentino Fernández
Marcus Frean
Richard Freeman
Toshio Fukuda
Colin Fyfe
Bogdan Gabrys
Marcus Gallagher
John Qiang Gan
Francisco Garijo
Mark Girolami
Antonio F. Gómez
Ana González
Angélica González
Manuel González
Daniel González
Francisco Herrera
Álvaro Herrero
James Hogan
Tony Holden
Jaakko Hollmen
Robert J. Howlett
David Hoyle
Hisao Ishibuchi
Paul Jackway
Gareth Jones
Vicente Julián
Ata Kaban
Hoon Kang
Juha Karhunen
Samuel Kaski
Dong Hwa Kim
Irwin King
Aldebaro Klautau
Mario Köppen
Kostadin Korutchev
Rudolf Kruse
Rosalía Laza
Kwong S. Leung
Carlos Linares
Paulo Lisboa
Eva Lorenzo
Frederic Maire
Roque Marín
José F. Martínez
Aitor Mata
José Ramón Méndez
Simon Miles

José Mira
José Manuel Molina
Carla Möller-Levet
Joaquín Pacheco
Juan Pavón
David Pelta
David Powers
José Principe
José Ramirez
Omer Rana
Vic Rayward-Smith
Perfecto Reguera
Bernadete Ribeiro
José Riquelme
Ramón Rizo
Roman Rosipal
Dymitr Ruta
Shazia Sadiq
José Santos
Michael Small
P.N. Suganthan
David Taniar
Dante Israel Tapia
Miguel Toro
Marcos Valiño
Marc Van Hulle
Alfredo Vellido
José Ramón Villar
Thomas Villmann
Lipo Wang
Dong-Qing Wei
Ian Wood
Gordon Wyeth
Yong Xu
Nobuyoshi Yabuki
Ronald R.Yager
Du Zhang
Yanqing Zhang
Ning Zhong
Rodolfo Zunino

Table of Contents

Learning and Information Processing

Data Mining and Information Management

Bioinformatics and Neuroinformatics

Agents and Distributed Systems

Financial Engineering and Modelling

Agent-Based Approach to Service Sciences

Neural-evolutionary Fusion Algorithms and Their Applications

Support Function Machines

Jiuzhen Liang

School of Information Technology, Jiangnan University
1800 Lihu Road, Wuxi, Jiangsu Province, CHINA 214122
jz.liang@yahoo.com.cn

Abstract. This paper proposes a novel model of support function machine (SFM) for time series predictions. Two machine learning models, namely, support vector machines (SVM) and procedural neural networks (PNN) are compared in solving time series and they inspire the creation of SFM. SFM aims to extend the support vectors to spatiotemporal domain, in which each component of vectors is a function with respect to time. In the view of the function, SFM transfers a vector function of time to a static vector. Similar to the SVM training procedure, the corresponding learning algorithm for SFM is presented, which is equivalent to solving a quadratic programming. Moreover, two practical examples are investigated and the experimental results illustrate the feasibility of SFM in modeling time series predictions.

Keywords: support vector machine, learning algorithm, support function, procedure neural networks, time series predictions.

1 Introduction

There has been more than ten years since support vector machines(SVM), the most popular machine learning model, was invented by V. Vapnik [1]. This decade discerned fast development of research on SVM both in theory and application. SVM as a general classifier or regression function has shown its great fascination than other models, such as neural networks, especially when samples are sparse and the established model by SVM has good(may be the best) generalization capability. SVM is recently of increasing interest more and more due to its promising empirical performance compared with other learning techniques [2]. Instead of using empirical risk minimization(ERM), which is commonly used in traditional learning, SVM is founded on structural risk minimization(SRM). ERM only minimizes the error occurred to training data whilst SRM minimizes an upper bound of the generalization error. This enables SVM to generalize well. The basic principle of SVM is to map the input space to a high-dimensional feature space using kernel techniques. A linear discriminant analysis is then formulated in the feature space to maximize the margin between two classes so as to maximize the generalization ability. Moreover, a discriminant analysis process is conducted based on a set of support vectors which are selected automatically from training data [3].

For time series predictions, SVM has been utilized as a regression function [4]. But while preparing samples for SVM, all functions which are dispersed in a certain interval of time, have to be transferred to spacial vectors. So essentially, SVM still performs

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1–9, 2007.

functions that map static vectors from one space to another. Recently, procedural neural networks (PNN) was proposed for spatiotemporal modeling [7]. PNN is a temporal neural networks model which aims to simulate time series predictions. Different from the classic neural networks (NN) in which neurons can not change state with respect to time, PNN combines the spatial and temporal information together, namely neurons process information both from space and time simultaneously. Based on PNN, some other models and learning strategies can be modified to simulate time series as well. In the past years, PNN models, properties and learning algorithms have been approached, such as complex number procedure neural networks [5], functional procedure neural networks [6], approximation ability to functional function [8] and trainings [9]. Naturally, the two models, SVM and PNN, inspire the following motivations: Can we find such a model, which shares the form with SVM, has the function similar to PNN and simulates time series better? Can we minimize the generalization error of PNN by introducing such support functions? This paper will investigate these problems and try to construct an efficient model for time series predictions.

The rest of this paper is organized as follows. In section two, a new model named support function machine (SFM) is established for simulating spatiotemporal problems. Section three deals with the learning algorithm for classification and regression respectively referring to the quadratic maximum problem of SVM, and for classification problems detail learning steps are presented. In section four, two examples, harm forecasts and stock predictions, are investigated. Finally, conclusions are given in section five.

2 Support Function Machines Model

Suppose we have N input patterns, $x_i \in \mathbb{R}^n$ be the i-th input pattern, where n is the number of the input variables, and y_i be the corresponding label of x_i. A SVM model based on the support vectors found through learning is defined as

$$f(\boldsymbol{x}, \alpha) = \sum_{i=1}^{N} \alpha_i y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + \alpha_0 \tag{1}$$

where, $K(\boldsymbol{x}, \boldsymbol{x}_i)$ is called a kernel function and $\alpha = (\alpha_0, \alpha_1, \cdots, \alpha_N)$ is the parameter vector needed to be confirmed.

In contrast to SVM, SFM is based on training data pairs $(\mathbf{x}_i(t), \mathbf{y}_i)$, in which $t \in [T_1, T_2]$, input patterns $\mathbf{x}_i(t) = (x_{i1}(t), x_{i2}(t), \cdots, x_{in}(t))$ and labels $\mathbf{y}_i = (y_{i1}, y_{i2}, \cdots, y_{im})$, where $i = 1, 2, \cdots, N$. Here m is the dimension of output, T_1 and T_2 are time boundary points. In this paper we only consider the case $n \geqslant 1, m = 1$ which corresponds to a multi-input and mono-output system. Different from SVM, all training data in SFM are vector functions of time which are discrete in interval $[T_1, T_2]$. In real world, sometimes data sampled in $[T_1, T_2]$ are not simultaneously recorded, i.e. in different dimensions different $t \in [T_1, T_2]$ are used as the sampling time points. This makes SVM difficult to simulate, because traditionally in SVM all the components of a vector must be captured simultaneously.

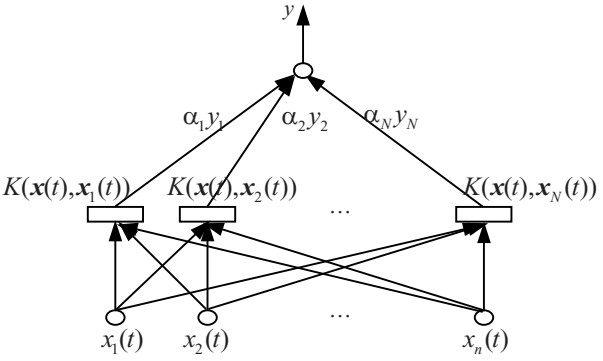


Fig. 1. Support function machine model

We denote $\boldsymbol{x}(t)$ as a general vector function and $\boldsymbol{x}_i(t)$ as the i-th support function (corresponding to the i-th support vector in SVM). Then a SFM classifier based on the support functions can be defined as

$$\mathbf{y} = f(\boldsymbol{x}(t), \alpha) = \sum_{i=1}^{N} \alpha_i y_i K(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) + \alpha_0 \tag{2}$$

which describes a functional relation between the vector function $\boldsymbol{x}(t)$ and the vector $\boldsymbol{y}$. The structure of SFM is much similar to SVM as in Fig. 1. Here, we omit the first subscripts of input vector $\boldsymbol{x}(t) = (x_1(t), x_2(t), \cdots, x_n(t))$ which denote the sequence numbers of samples.

A major revision is then made to determine the kernel function using functional similarity for recognizing different attributes. The kernel functional function of the i-th support function is defined as

$$K(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) = \exp(\beta \frac{s(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) - b_i}{b_i}) \tag{3}$$

where $s(\boldsymbol{x}(t), \boldsymbol{x}_i(t))$ is a pair-wise similarity between function $\boldsymbol{x}(t)$ and $\boldsymbol{x}_i(t)$, b_i is the maximum similarity associated with the i-th support function and β is a constant. It can be seen that the term in the exponent function, which can be denoted as

$$c(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) = \frac{s(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) - b_i}{b_i} \tag{4}$$

is in general non-positive.

In this study, we revise the form of bio-SVM [3] and promote SFM, which can be regarded as the generalization of bio-SVM. Actually, there are many ways to define the similarity of two functions and each similarity corresponds to a distance between two functions, for example, in Euclid space

$$s(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) = \frac{1}{T_1 - T_2} \int_{T_1}^{T_2} \frac{\boldsymbol{x}(t) \cdot \boldsymbol{x}_i(t)}{\|\boldsymbol{x}(t)\| \|\boldsymbol{x}_i(t)\|} \mathrm{d}t \tag{5}$$

in which '$\cdot$' denotes the inter-product of two vectors, $\|\boldsymbol{x}(t)\|$ denotes a norm of the vector, while $\boldsymbol{x}(t)$ and $\boldsymbol{x}_i(t)$ are continuous in $[T_1, T_2]$. If $\boldsymbol{x}(t)$ is a non-numerical series function, for instance, a sequence of discrete data $\boldsymbol{x}(t)|_{t\in[T_1,T_2]} = \{x_{ljk}|j = 1, \cdots, n; k = 1, \cdots, n_j\}$ and $\boldsymbol{x}_i(t) = \{x_{ijk}|j = 1, \cdots, n; k = 1, \cdots, n_j\}$, here, $l, i = 1, \cdots, N$ and n_j is the hits number of component $x_{ij}(t)$ in $[T_1, T_2]$, the similarity can be defined as

$$s(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) = \frac{\|\boldsymbol{x}(t) \cap \boldsymbol{x}_i(t)\|}{\|\boldsymbol{x}(t) \cup \boldsymbol{x}_i(t)\|} \tag{6}$$

where $\|A\|$ denotes the size of the set A.

In this paper, for the case of numerical samples in the Euclid space and $\boldsymbol{x}_i(t_k) = \{x_{ijk}|j = 1, \cdots, n\}$, we define the corresponding vector function similarity as

$$s(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) = \frac{1}{M} \sum_{k=1}^{M} \frac{\boldsymbol{x}(t_k) \cdot \boldsymbol{x}_i(t_k)}{\|\boldsymbol{x}(t_k)\| \|\boldsymbol{x}_i(t_k)\|} \tag{7}$$

in which M denotes the sampling scale in the temporal domain.

Unfortunately, the similarity definition can not guarantee the corresponding kernel function to be semidefinite positive, which is a Mercer's kernel. In fact, it is well known that the exponential of a distance is not, in general, a Mercer's kernel. However, the proposed kernel can always be made semidefinite positive by appropriate choice of parameter (β as in (3)) [14]. In particular, as long as no two samples in the training set are exactly alike, it is always possible to make the kernel matrix diagonally dominant, and therefore semidefinite positive, by making β sufficiently large. Therefore, the positive definiteness of the kernel can usually be checked by evaluating the positive definiteness of the kernel matrix obtained with the data sets.

3 Learning Algorithm

Typically, a learning algorithm for SVM is equivalent to solving a quadratic maximum or minimum problem. This strategy is also valid for SFM except that it is different in computing the kernel function. In the case of classification, the primal objective function (which should be minimized) is

$$E(\alpha) = \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i(t), \boldsymbol{x}_j(t)) - \sum_{i=1}^{N} \alpha_i \tag{8}$$

subject to the box constraint

$$0 \le \alpha_i \le C, \forall i \tag{9}$$

and the linear constraint

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{10}$$

Here C is the restriction bound and can be different in each loop of computing α. In the process of searching optimal α_i, for instance, in the s-th iteration, it takes the form as

$$C(s) = \max_i\{\alpha_i(s)\} + 1 \tag{11}$$

This is a quadratic restrictive optimal problem with respect to α_i and can be solved by a gradient descend strategy, i.e. update α by

$$\alpha_i(s+1) = \min\{C(s), \max\{0, \alpha_i(s) - \Delta\alpha_i(s)\}\} \tag{12}$$

where

$$\Delta\alpha_i(s) = \eta(y_i \sum_{j=1}^{N} \alpha_j(s) y_j K(\boldsymbol{x}_i(t), \boldsymbol{x}_j(t)) - 1) \tag{13}$$

Here, η is learning rate, $\eta \in (0, 1)$, and typically η is fixed.

While for regression functional function, similar to [10] SFM minimizes functional function as follows.

$$E(\alpha) = \frac{1}{N} \sum_{i=1}^{N} |y_i - f(\boldsymbol{x}_i(t), \alpha)|_\varepsilon + \|\alpha\|^2, \tag{14}$$

where $|x|_\varepsilon$ is an ε-insensitive error function defined as

$$|x| = \begin{cases} 0, & \text{if } |x| < \varepsilon \\ |x| - \varepsilon, & \text{otherwise} \end{cases} \tag{15}$$

and the output of SFM has the following form

$$f(\boldsymbol{x}(t), \alpha^*, \alpha) = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) K(\boldsymbol{x}(t), \boldsymbol{x}_i(t)) + \alpha_0 \tag{16}$$

Intuitively, α_i^* and α_i are "positive" and "negative" Lagrange multipliers (i.e., a single weight) that obey $\alpha_i^*, \alpha_i \geq 0, \forall i$, and $\sum_{i=1}^{N}(\alpha_i^* - \alpha_i) = 0$.

Referring to the dual theorem [11], the primal form of equation (16) can be written as

$$\begin{aligned} L(\alpha^*, \alpha) =& \varepsilon \sum_{i=1}^{N} (\alpha_i^* + \alpha_i) - \sum_{i=1}^{N} y_i(\alpha_i^* - \alpha_i) \\ &+ \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K(\boldsymbol{x}_j(t), \boldsymbol{x}_i(t)), \end{aligned} \tag{17}$$

where the objective function should be minimized with respect to α^* and α, subject to the constraints

$$\begin{cases} \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0 \\ 0 \leq \alpha_i^*, \alpha_i \leq C, \forall i, \end{cases} \tag{18}$$

here the parameter C is the same user-defined constant that represents a balance between the model complexity and the approximation error.

For the case of classification, the algorithm can be described in detail as follows.

Step 1. Initialize all parameters including SFM structure parameters (e.g. α) and learning algorithm parameters (e.g. error precision ε).

Step 2. Choose an appropriate β and compute the kernel matrix by equation (3) until it is semidefinite positive.

Step 3. Compute error function E_1 according to equation (8) for all training samples.

Step 4. Update parameters α by equation (12) and equation (13).

Step 5. Compute error function E_2 by equation (8).

Step 6. If $|E_2 - E_1| < \varepsilon$ stop, otherwise let $E_1 = E_2$, go to step 3.

4 Application Examples

4.1 Harm Forecast of Horsetail Pine Worms

Horsetail pine worm is one of major harms to the forest in the southeast China. Although much more has been done in development of techniques to protect forest source, it is necessary to forecast the degree of worm harm in the coming season. The data is provided by the Institute of Jinhua Epidemic Prevention. Each record includes four segments, namely, later age through winter(LA), the first age(FA), the second age(SA), and the anterior age through winter(AA). For each segment there are eight observation fields including light-degree harm (LDH), middle-degree harm (MDH), heavy-degree harm (HDH), total amount of worms(TAW), average amount of worm in one tree (AAW), average amount of tree worm-harmed (AAT), area worm-harmed (AWH) and area of one degree worm-harmed (AOD). In Table 1 we just give two records of the horsetail worm harm affection from 1995 to 1996. Data in one year naturally form one record corresponding to one sample for training SFM. In each sample the four segments correspond to four values for each field in the temporal domain. The seven columns of values (except HDH) compose the input vectors for SFM, and the labels of heavy degree harm (HDH) in the associated seasons are the desire outputs of SFM (1 denotes heavy and -1 denotes not heavy).

Table 1. Two samples of horsetail worm harm

Segments	LDH	MDH	HDH	TAW	AAW	AAT	AWH	AOD
LA	22636	899	-1	23535	2.1	42.2	178581	155046
FA	64798	17867	-1	82665	3.5	37.8	262990	180325
SA	113212	50202	1	163414	3.9	42.3	388334	224920
AA	154515	36068	1	192430	4.5	47.1	411479	219049
LA	115497	13665	1	129162	4.2	47.0	353690	224528
FA	147156	27869	1	175025	4.4	57.6	445607	270582
SA	125226	21048	1	146274	3.0	50.0	435536	289262
AA	63182	280	-1	63462	2.4	53.4	313313	249851

Table 2. Warm harm heavy degree predictions

Training samples	Test samples	Error samples	Accuracy
20	19	10	47.37 %
26	13	4	69.23 %
30	9	2	77.78 %
32	7	1	85.71 %
34	5	0	100 %

Totally 39 samples are used in this experiment including training and test. Five groups of data are designed and in each group data is randomly divided into two sets: training and test. The experiment results are summarized in table 2.

4.2 Stock Price Predictions

It is a practically interesting and challenging topic to predict the trends of a stock price. Fundamental and technical analysis were the first two methods used to forecast stock prices. Various technical, fundamental, and statistical indicators have been proposed and used with varying results. However, no one technique or combination of techniques has been successful enough to consistently ”beat the market”[12]. Much more work has been done on stock markets predictions. We do not discuss more theories on stock market here, and we regard it as a typical example of time series. The stock data comes from Yahoo finance web site[13] in the period from 01/01/2000 to 06/30/2001. Data in one week or five days composes a sample of SFM. For each sample there are five observation fields including the open price, the highest price, the lowest price, the closing price and the stock volume. Table 3 lists two samples of Yahoo stock from 01/03/2000 to 01/14/2000.

In this experiment we choose 100 samples continuously from the data list, and for each prediction we select 20 samples, which are the closest to the prediction one in date, in training SFM. Each sample is composed of data from five sequential days. The prediction and actual values for open price are plotted in Fig. 2, and so are for close price in Fig. 3.

Table 3. Ten records of a stock price list

Date	Open	High	Low	Close	Volume
01/03/2000	153.00	153.69	149.19	150.00	22069800
01/04/2000	147.25	148.00	144.00	144.00	22121400
01/05/2000	143.75	147.00	142.56	143.75	27292800
01/06/2000	143.13	146.94	142.63	145.67	19873200
01/07/2000	148.00	151.88	147.00	151.31	20141400
01/10/2000	152.69	154.06	151.13	151.25	15226500
01/11/2000	151.00	152.69	150.63	151.50	15123000
01/12/2000	151.06	153.25	150.56	152.00	18342300
01/13/2000	153.13	154.94	153.00	153.75	14953500
01/14/2000	153.38	154.63	149.56	151.00	18480300

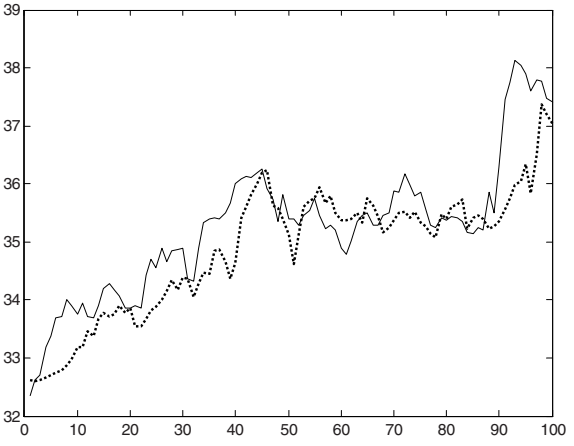


Fig. 2. Predictive and actual value of open price. Here the solid curve denotes the actual price of the stock, the dotted curve indicates the prediction value, the horizontal direction shows the date and the vertical direction presents the stock price.

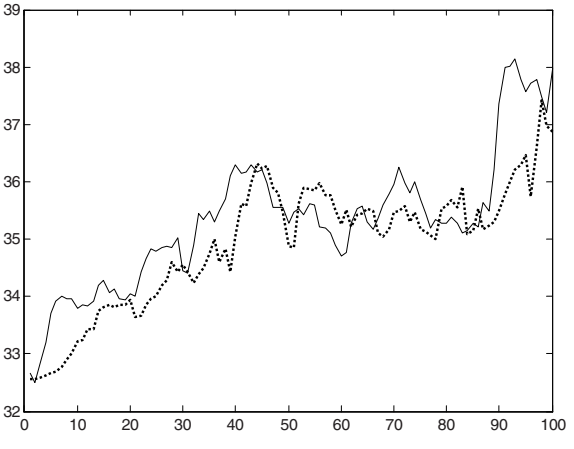


Fig. 3. Predictive and actual value of close price. The meanings of the two curves in this figure are the same as in Fig.2. Here 100 records of daily stock prices are investigated as the test set of SFM model. For the sake of intuition, all the discrete points are connected with line and each of the curve tendency is plotted clearly.

5 Conclusions

The purpose of this paper is to generalize SVM and promote SFM in which input patterns are functions of time. In the case of classifications, sometimes we have to deal with such problems that need to separate vector functions in a function space. While the task of function regression is to simulate time series in a spatiotemporal domain.

The main contribution of this paper is to define the similarity between two vector functions and give the kernel function of vector functions. The learning algorithm for SFM is in no discrimination with that of the traditional SVM except the kernel functions. In real world, there are many problems associated with a procedure or varied with respect to time. SFM provides a new attempt to model such time series issues.

Acknowledgement. This work is partly supported by CMP laboratory, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University. The author is supported by the Agreement between Czech Ministry of Education and Chinese Ministry of Education. Thanks to Pavol Vlcek for his suggestions on revision. Thanks to the session chair and two referees for their very valuable comments and suggestions.

References

1. Vapnik, V.: The nature of statistical learning theory. Springer, New York (1995)
2. Scholkopf, B., Sung, K.K., Burges, C.J.C., Girosi, F., Niyogi, P., Poggio, T., Vapnik, V.: Comparing support vector machines with Gaussian kernels to radial basis function classifiers. IEEE Trans. Signal Process. 45, 2758–2765 (1997)
3. Yang, Z.R., Chou, K.-C.: Bio-support vector machines for computational proteomics. Bioinformatics 20, 735–741 (2004)
4. Vapnik, V.: The support vector method of function estimation. In: Suykens, J.A.K., Vandewalle, J. (eds.) Nonlinear Modeling: Advanced Black-Box Techniques, pp. 55–85. Kluwer, Boston, MA (1998)
5. Liang, J.Z, Han, J.M.: Complex number procedure neural networks. In: Wang, L., Chen, K., Ong, Y.S. (eds.) ICNC 2005. LNCS, vol. 3610, pp. 336–339. Springer, Heidelberg (2005)
6. Liang, J.Z.: Functional procedure neural networks. Dynamic of Continuous Discrete and Impulsive Systems-Series B-Applications & Algorithms 1 (S.I.), pp. 27–31 (2005)
7. Liang, J.Z., Zhou, J.Q., He, X.G.: Procedure neural networks with supervised learning. In: The 9th International Conference on Neural Information Processing, Singapore, pp. 523–527 (2002)
8. He, X.G., Liang, J.Z.: Some theoretic problems of procedure neural networks. Engineering Science in China 2, 40–44 (2000)
9. He, X.G., Liang, J.Z., Xu, S.H.: Training and application of procedure neural networks. Engineering Science in China 3, 31–35 (2001)
10. Flake, G., Lawrence, S.: Efficient SVM regression training with SMO. Machine Learning 46, 271–290 (2002)
11. Nello, C., John, S.T.: An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge (2000)
12. Lawrence, R.: Using neural networks to forecast stock market prices. Reports (1997), http://people.ok.ubc.ca/rlawrenc/research/Papers/nn.pdf
13. http://finance.yahoo.com/q/hp?s=GE&a=00&b=1&c=2007&d=06&e=26&f=2007&g=d (2007)
14. Chan, A., Vasconcelos, N., Moreno, P.J.: A family of probabilistic kernels based on information divergence. Technical Report SVCL-TR-2004-01 (June 2004)

Different Bayesian Network Models in the Classification of Remote Sensing Images

Cristina Solares* and Ana Maria Sanz

University of Castilla-La Mancha, Spain
Cristina.Solares@uclm.es
Ana.Sanz@uclm.es

Abstract. In this paper we study the application of Bayesian network models to classify multispectral and hyperspectral remote sensing images. Different models of Bayesian networks as: Naive Bayes (NB), Tree Augmented Naive Bayes (TAN) and General Bayesian Networks (GBN), are applied to the classification of hyperspectral data. In addition, several Bayesian multi-net models: TAN multi-net, GBN multi-net and the model developed by Gurwicz and Lerner, TAN-Based Bayesian Class-Matched multi-net (tBCM^2) (see [1]) are applied to the classification of multispectral data. A comparison of the results obtained with the different classifiers is done.

1 Introduction

Classification problems (see [2]) occur in a wide range of situations in real life such as disease diagnosis, image recognition, fault diagnosis, etc.

Probabilistic models, especially those associated with Bayesian networks, are very popular as a formalism for handling uncertainty. The increasing number of applications developed these last years show that this formalism has practical value also.

In this paper we apply different models of Bayesian networks to the classification of remote sensing images, considering multispectral and hyperspectral data sets. In a multispectral image the number of spectral bands for each pixel is less than 20, otherwise the image is called hyperspectral.

The paper is organized as follows. Section 2 introduces the Bayesian networks and the Bayesian networks as classifiers. Six models of Bayesian networks are introduced: General Bayesian network(GBN), Naive Bayes (NB), Tree Augmented Naive Bayes (TAN), TAN Bayesian multi-net, GBN Bayesian multi-net and the TAN-Based Bayesian Class-Matched multi-net (tBCM^2). Section 3 presents the application of the above models to the classification of remote sensing images. In Sect. 4 some conclusions are given.

* The author is indebted to the Spanish Ministry of Science and Technology (Project MTM2006-15671) and to the Junta de Comunidades de Castilla-La Mancha (Project PAI-05-044) for partial support.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 10–16, 2007.

2 Bayesian Networks

A Bayesian Network (BN) (see [3] for further details) over $\boldsymbol{X} = (X_1, \ldots, X_n)$ is a pair (D, P), where D is a directed acyclic graph with one node for each variable in $\boldsymbol{X}$ and $P = \{p_1(x_1|\pi_1), \ldots, p_n(x_n|\pi_n)\}$ is a set of n conditional probability distributions, one for each variable, given the values of the variables on its parent set Π_i (CP table). Each node in D represents a domain variable (eg, a dataset attribute) and each arc in D represents a probabilistic dependence between two variables quantified using the above CP table.

Here x_i and π_i denote realizations (instantiations) of X_i and Π_i, respectively. The joint probability distribution (JPD) of $\boldsymbol{X}$ can then be written as

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p_i(x_i|\pi_i) \ . \tag{1}$$

2.1 Bayesian Network Classifiers

The application of Bayesian network models to classification involves two subtasks: Learning the BN structure (the graphical structure D) and the BN parameters (CP table). It is trivial to learn the parameters for a given structure, simply use the empirical conditional frequencies from the data (see [4]). Constructing the BN structure can be performed using expert knowledge or directly from the data. There are different methods of learning a BN structure, as the score-based methods (see [4]) and the methods that learn the structure by identifying the conditional independence relation-ships among the nodes (CI-based methods). The score-based methods incorporate a search procedure to find a network structure and a score is employed to evaluate each structure in the search space. The K2 algorithm, introduced in [4], is a search algorithm for finding a high quality Bayesian network in a reasonable time. An example of CI-based method is the algorithm described in Cheng et al. [5]. Cheng et al. in [6] show that the CI-based learning algorithms are very efficient and the learned BN classifiers can give very good prediction accuracy. Next we describe the different models of Bayesian network classifiers used in this paper.

GeneralBayesian Network (GBN). A GBN with JPD $p(a_1, a_2, \ldots, a_n, c)$ defined as in (1), can be constructed to solve a classification problem (see Fig. 1). The variables $\boldsymbol{A} = (A_1, \ldots, A_n)$ are the attributes of the problem and C is the class variable having k different states. The resulting model can be used to classify a given set of attribute values $\boldsymbol{a} = (a_1, \ldots, a_n)$ (see [7]). The vector $\mathbf{a}$ belongs to the class $c \in C$ that maximizes the posterior probability $p(c|\mathbf{a})$. The structure of the GBN can be learned using a score-based method as the K2 algorithm (see [4]) or a CI based method as the algorithm introduced in [5]. In this paper, we use the K2 search algorithm.

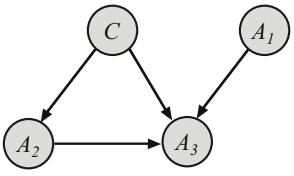


Fig. 1. Example of General Bayesian Network (GBN). C is the class variable and A_1, A_2, A_3 are the attribute variables.

Naive Bayes (NB). A NB is a simple structure of Bayesian network, the class node C is a parent of all other nodes (attributes) and there are not other connections between the nodes (see [7]).

Tree Augmented Naive Bayes (TAN). The very strong assumption of independence of all the attributes given its parents set in the Naive Bayes, is relaxed in this type of network. The TAN algorithm constructs a tree structure between the attribute nodes and after that adds a link from the class node C to the attribute nodes $A_i, i = 1, \ldots, n$ (see [7]). This model is based in the algorithm described by Chow et al. in [8], for learning tree-like Bayesian networks.

GBN Bayesian Multi-net. A GBN Bayesian multi-net is a generalization of the GBN, a different GBN is built for each class value and a set of networks is used as a classifier (see Fig. 2). For that, we partition the training data set by classes and for each class value we construct a GBN for the attribute variables.

TAN Bayesian Multi-net. In the TAN model, the relations between the attributes are the same for all the different instances of the class variable C. A Bayesian TAN multi-net is a generalization, a different TAN is built for each class value and a set of networks is used as a classifier (see [7]). This model allows the relations among the attributes to be different for the different values of the class. For that, we partition the training data set by classes and for each class value we construct a TAN for the attribute variables.

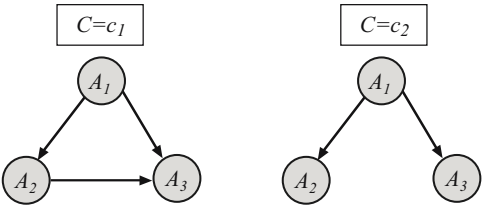


Fig. 2. Example of Bayesian Multi-net. C is the class variable that takes two values c_1 and c_2, and A_1, A_2, A_3 are the attribute variables.

TAN-Based Bayesian Class-Matched Multi-net (tBCM^2). The tBCM^2 is a multi-net classifier that learns each local network (BN associated to each class value) using a detection-rejection measure (see [1]). The algorithm searches for the structure maximizing a discrimination-driven score that is calculated using training data for all the classes. The structure of each local network in tBCM^2 is based on the TAN model and it is learned using the SuperParent algorithm (see Keogh et al. in [9]). In Gurwicz et al.[1] the average superiority of the tBCM model in comparison with other classifiers, as the TAN multi-net, is shown.

In the next section the above models of Bayesian network classifiers are applied to the classification of remote sensing images.

3 Remote Sensing Image Classification

The models of Bayesian networks introduced in Sect. 2.1 can be applied to classify remote sensing spectral images. For the implementation of the proposed models, we use the Bayes Net toolbox in matlab (see [10]) and the BNT Structure Learning Package (see [11]).

A remote sensing spectral image consists of an array of multidimensional vectors assigned to particular spatial regions (pixel locations), reflecting the response of a spectral sensor at various wavelengths. Formally these images can be described as a matrix $\boldsymbol{V} \equiv (\boldsymbol{v}_{11}(x^1, y^1), \ldots, \boldsymbol{v}_{nm}(x^n, y^m))$ where $\boldsymbol{v}_{ij}(x^i, y^j) \in R^l, i = 1, \ldots, n, j = 1, \ldots, m$ is the vector of spectral information associated with pixel location (x^i, y^j) and the vector components $v_{ijk}(x^i, y^j), k = 1, \ldots, l$ reflects the responses of a spectral sensor at various wavelengths.

In this application all variables (class variable and attributes of the problem) are assumed to be discrete, that is, each variable has a finite set of possible values.

3.1 An Example of Multispectral Data Set Analysis

In the present contribution we consider a LANDSAT TM image from Sierra de Gredos (Spain). This image has been obtained from the GIS IDRISI 32 tutorial (http://www.clarklabs.org/). LANDSAT TM satellite-based sensors produce images of the Earth in different spectral bands. In this work six bands (bands 1-5 and band 7) are strategically determined for optimal detection and discrimination of water, soil and four different forest type, these are the class values for the classification problem. Band 6 is often dropped from analysis because of the lower spatial resolution. The spectral information, associated with each pixel of a LANDSAT scene is represented by a vector $\boldsymbol{v}(x, y) \in R^6$, these vectors are the attribute values of the problem. This is a classification problem with six attributes and six class values.

The GBN, NB and TAN models, have been previously applied by the authors, to the analysis of a multispectral data (see [12]). A GBN multi-net model also has been previously applied to this problem (see Ouyang et al. in [13]). In this paper, we apply the GBN multi-net, TAN multi-net and the tBCM^2 multi-net models to the classification of multispectral remote sensing images.

Table 1. Training and test accuracy (mean (±std) in %) obtained with each classifier in Sect. 3.1

Classifier	Training	Test
TAN Multi-net	84.38(±0.11)	84.29(±0.40)
GBN Multi-net	84.84(±0.19)	84.47(±0.68)
tBCM^2 Multi-net	85.50(±0.12)	85.20(±0.45)

The above classification problem is analyzed using 5-fold cross-validation (CV5). We apply the different models of Bayesian multi-net classifiers to classify the multispectral image, the training and test accuracy obtained are shown in Table 1. A comparison shows negligible differences between the TAN and the GBN Bayesian multi-net models. All the models obtain almost 85% of accuracy with a slight advantage of the tBCM^2 multi-net.

3.2 An Example of Hyperspectral Data Set Analysis

For some years, the above application has been limited to data of low dimensionality, less than 10 bands (multispectral data). Recent advances in sensor technology make possible to work with several hundred bands (hyperspectral data). In this paper, we do the novel application of the NB, GBN and TAN models to the classification of hyperspectral data. The hyperspectral data used in our experiments is a section of a scene taken over northwest Indiana's Pines by the AVIRIS sensor in 1992 (ftp://ftp.ecn.purdue.edu/biehl/MultiSpec/). The AVIRIS sensor collects 224 bands of data but four of these bands contain only zeros and consequently they are eliminated. The initial 220 bands are reduced to 200 because the bands covering the region of water absorption: $[104-108], [150-163], 220$ are removed. In this work, 200 bands are considered for optimal detection and discrimination of 9 different classes: Corn-no till, Corn-min till, Grass/Pasture, Grass/Trees, Hay-windrowed, Soybean-no till, Soybean-min till, Soybean-clean till and Woods. From the initial 16 land-cover classes, seven were eliminated, since only few training samples were available for them. The above is a classification problem with 200 attributes and 9 class values.

We analyze the effectiveness of Bayesian networks in classifying hyperspectral images directly in the original hyperdimensional attribute space. The problem is studied using 5-fold cross-validation (CV5). We apply the different models of Bayesian network classifiers (NB, TAN and GBN) to the above classification problem, the training and test accuracy obtained are shown in Table 2. A comparison shows slight differences between the TAN and GBN Bayesian network models, both are superior on accuracy to the NB model. The very strong assumption of independence of all the attributes given its parents set in the NB model is not realistic in the case of study.

Table 2. Training and test accuracy (mean (±std) in %) obtained with each classifier in Sect. 3.2

Classifier	Training	Test
NB	58(±0.24)	58(±0.20)
TAN	88(±0.23)	80(±0.58)
GBN	84(±0.34)	80(±0.80)

4 Conclusions

Bayesian networks appear as powerful tools in hyperspectral remote sensing image classification. Different models of Bayesian networks as: Naive Bayes (NB), Tree Augmented Naive Bayes (TAN) and General Bayesian Network (GBN), have been applied to the classification of an hyperspectral image. In addition, several Bayesian multi-net models: TAN multi-net, GBN multi-net and the model developed by Gurwicz and Lerner, TAN-Based Class-Matched multi-net ($tBCM^2$) are applied to the classification of multispectral data. Feature (attribute) selection is an important task in remote sensing data processing, particularly in case of hyperspectral images. Actually, we are studying the application of Bayesian network models to the classification of hyperspectral data, combined with a band selection method to reduce the dimensionality of the feature space.

References

1. Gurwicz, Y., Lerner, B.: Bayesian class-matched multinet classifier. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) SSPR 2006. LNCS, vol. 4109, pp. 145–153. Springer, Heidelberg (2006)
2. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley & Sons, New York (2001)
3. Castillo, C., Gutiérrez, J.M., Hadi, A.S.: Expert Systems and Probabilistic Network Models. Springer, New York (1997)
4. Cooper, G.F., Herskovitz, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–347 (1992)
5. Cheng, J., Bell, D.A., Liu, W.: An algorithm for Bayesian belief network construction from data. In: Proc. AI & STAT 1997, pp. 83–90 (1997)
6. Cheng, J., Greiner, R.: Learning Bayesian belief network classifiers: Algorithms and system. In: Proc. 14th Canadian Conf. on Artificial Intelligence, pp. 141–151 (2001)
7. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29, 131–163 (1997)
8. Chow, C., Liu, C.: Approximating discrete probability distributions with dependence trees. IEEE Trans. Inf. Theory 14(3), 462–467 (1968)
9. Keogh, E.J., Pazzani, M.J.: Learning the structure of augmented Bayesian classifiers. Int. J. on Artificial Ingelligence Tools 11(4), 587–601 (2002)

10. Murphy, K.P.: The Bayes Net Toolbox for matlab. Computing Science and Statistics 33 (2001)
11. Leray, P., Francois, O.: BNT, Structure learning package: documentation and experiments. Technical Report. Laboratoire PSI-INSA Rouen-FRE CNRS 2645 (2004)
12. Solares, C., Sanz, A.M.: Bayesian network classifiers. Some engineering applications. In: Proc. 9th IASTED Int. Conf. Artificial Intelligence and Soft Computing, pp. 331–335 (2005)
13. Ouyang, Y., Ma, J., Dai, Q.: Bayesian multinet classifier for classification of remote sensing data. Int. J. of Remote Sensing 27, 4943–4961 (2006)

Group Decision Making with Triangular Fuzzy Linguistic Variables

Zeshui Xu

Department of Management Science and Engineering
School of Economics and Management
Tsinghua University, Beijing 100084, China
Xu_zeshui@263.net

Abstract. In group decision making with linguistic information, the decision makers (DMs) usually provide their assessment information by means of linguistic variables. In some situations, however, the DMs may provide fuzzy linguistic information because of time pressure, lack of knowledge, and their limited attention and information processing capabilities. In this paper, we introduce the concepts of triangular fuzzy linguistic variable and its member function, and introduce some operational laws of triangular fuzzy linguistic variables. We propose a formula for comparing triangular fuzzy linguistic variables, and develop some operators for aggregating triangular fuzzy linguistic variables, such as the fuzzy linguistic averaging (FLA) operator, fuzzy linguistic weighted averaging (FLWA) operator, fuzzy linguistic ordered weighted averaging (FLOWA) operator, and induced FLOWA (IFLOWA) operator, etc. Based on the FLWA and IFLOWA operators, we develop a practical method for group decision making with triangular fuzzy linguistic variables, and finally, an illustrative example is given to verify the feasibility and effectiveness of the developed method.

1 Introduction

Group decision making with linguistic information involves aggregating different individual linguistic decision information into the collective linguistic decision information, which is a hot research topic having received more and more attention from researchers [1-14]. Herrera et al. [1-6], Bordogna et al. [7], Marimin et al. [8], and Xu [9-14] have investigated the group decision making problem, in which the decision information is expressed as linguistic variables. In some situations, however, the DMs may provide fuzzy linguistic information because of time pressure, lack of knowledge, and their limited attention and information processing capabilities. In this paper, we shall investigate another kind of group decision making problem, in which the DMs can only provide their preferences in the form of triangular fuzzy linguistic variables. To do so, the remainder of this paper is structured as follows. In Section 2 we define the concept of triangular fuzzy linguistic variable and some operational laws of triangular fuzzy linguistic variables, and propose a formula for comparing triangular fuzzy linguistic variables. In Section 3 we propose some operators for aggregating triangular fuzzy linguistic variables. Section 4 develops a practical method for group decision making with triangular fuzzy linguistic variables. Section 5 gives an illustrative example, and Section 6 concludes this paper.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 17–26, 2007.

2 Triangular Fuzzy Linguistic Variables

In decision making with linguistic information, the DM generally provides his/her assessment information by using linguistic scale [1-14]. Suppose that $S = \{s_i \mid i = -t,...,t\}$ is a finite and totally ordered discrete label set. Any label, s_i, represents a possible value for a linguistic variable, and it requires that $s_i < s_j$ iff $i < j$. For example, a label set S could be [12]:

$$S = \{s_{-4} = extremely\ \ poor,\ s_{-3} = very\ \ poor,\ s_{-2} = poor,$$
$$s_{-1} = slightly\ \ poor,\ s_0 = fair,\ s_1 = slightly\ \ good,$$
$$s_2 = good,\ s_3 = very\ good,\ s_4 = extremely\ good\}$$

In the process of aggregating information, some results may do not exactly match any linguistic labels in S. To preserve all the given information, Xu [12] extended the discrete label set S to a continuous label set $\overline{S} = \{s_\alpha \mid \alpha \in [-q, q]\}$, where $q\ (q > t)$ is a sufficiently large positive integer. If $s_\alpha \in S$, then s_α is called an original linguistic label, otherwise, s_α is called a virtual linguistic label. In general, the DMs use the original linguistic labels to evaluate alternatives, and the virtual linguistic labels can only appear in operation.

Definition 2.1. *Let $s_\alpha, s_\beta \in \overline{S}$, then we define the distance between s_α and s_β as:*

$$d(s_\alpha, s_\beta) = |\alpha - \beta| \tag{1}$$

In some situations, however, the DMs may provide fuzzy linguistic information because of time pressure, lack of knowledge, and their limited attention and information processing capabilities, in the following, we define the concept of triangular fuzzy linguistic variable.

Definition 2.2. *Let $\hat{s} = (s_\alpha, s_\beta, s_\gamma) \in \hat{S}$, where s_α, s_β and s_γ are the lower, modal and upper values of $\hat{s}$, respectively, then we call $\hat{s}$ a triangular fuzzy linguistic variable, which is characterized by the following member function:*

$$\mu_{\hat{s}}(\theta) = \begin{cases} 0, & s_{-q} \le s_\theta \le s_\alpha \\ \dfrac{d(s_\theta, s_\alpha)}{d(s_\beta, s_\alpha)}, & s_\alpha \le s_\theta \le s_\beta \\ \dfrac{d(s_\theta, s_\gamma)}{d(s_\beta, s_\gamma)}, & s_\beta \le s_\theta \le s_\gamma \\ 0, & s_\gamma \le s_\theta \le s_q \end{cases} \tag{2}$$

Clearly, s_β gives the maximal grade of $\mu_{\hat{s}}(\theta)$ ($\mu_{\hat{s}}(\theta) = 1$), s_α and s_γ are the lower and upper bounds which limit the field of the possible evaluation. Especially, if $s_\alpha = s_\beta = s_\gamma$, then $\hat{s}$ is reduced to a linguistic variable.

Let $\hat{S}$ be the set of all triangular fuzzy linguistic variables. Consider any three triangular fuzzy linguistic variables $\hat{s} = (s_\alpha, s_\beta, s_\gamma)$, $\hat{s}_1 = (s_{\alpha_1}, s_{\beta_1}, s_{\gamma_1})$, $\hat{s}_2 = (s_{\alpha_2}, s_{\beta_2}, s_{\gamma_2})$, and $\lambda \in [0,1]$, then we define their operational laws as follows:

1) $\lambda \hat{s} = \lambda(s_\alpha, s_\beta, s_\gamma) = (\lambda s_\alpha, \lambda s_\beta, \lambda s_\gamma) = (s_{\lambda\alpha}, s_{\lambda\beta}, s_{\lambda\gamma})$;

2) $\hat{s}_1 \oplus \hat{s}_2 = (s_{\alpha_1}, s_{\beta_1}, s_{\gamma_1}) \oplus (s_{\alpha_2}, s_{\beta_2}, s_{\gamma_2}) = (s_{\alpha_1} \oplus s_{\alpha_2}, s_{\beta_1} \oplus s_{\beta_2}, s_{\gamma_1} \oplus s_{\gamma_2})$
$= (s_{\alpha_1+\alpha_2}, s_{\beta_1+\beta_2}, s_{\gamma_1+\gamma_2})$;

3) $\lambda(\hat{s}_1 \oplus \hat{s}_2) = \lambda \hat{s}_1 \oplus \lambda \hat{s}_2$;

4) $\hat{s}_1 \oplus \hat{s}_2 = \hat{s}_2 \oplus \hat{s}_1$.

In the following, we introduce a formula for comparing triangular fuzzy linguistic variables:

Definition 2.3. *Let* $\hat{s}_1 = (s_{\alpha_1}, s_{\beta_1}, s_{\gamma_1})$, $\hat{s}_2 = (s_{\alpha_2}, s_{\beta_2}, s_{\gamma_2}) \in \hat{S}$, *then the degree of possibility of* $\hat{s}_1 \geq \hat{s}_2$ *is defined as:*

$$p(\hat{s}_1 \geq \hat{s}_2) = \rho \max\left\{1 - \max\left(\frac{d(s_{\beta_2}, s_{\alpha_1})}{d(s_{\beta_1}, s_{\alpha_1}) + d(s_{\beta_2}, s_{\alpha_2})}, 0\right), 0\right\}$$
$$+ (1-\rho) \max\left\{1 - \max\left(\frac{d(s_{\gamma_2}, s_{\beta_1})}{d(s_{\gamma_1}, s_{\beta_1}) + d(s_{\gamma_2}, s_{\beta_2})}, 0\right), 0\right\} \tag{3}$$

where the value ρ *is an index of rating attitude. It reflects the DM's risk-bearing attitude. If* $\rho < 0.5$, *then the DM is a risk lover. If* $\rho = 0.5$, *then the DM's attitude is neutral to the risk. If* $\rho > 0.5$, *the DM is a risk avertor. Especially, if both the linguistic variables* $\hat{s}_1$ *and* $\hat{s}_2$ *express precise linguistic information (i.e. if*

$$d(s_{\gamma_1}, s_{\alpha_1}) + d(s_{\gamma_2}, s_{\alpha_2}) = 0 \tag{4}$$

then we define the degree of possibility of $\hat{s}_1 > \hat{s}_2$ *as:*

$$p(\hat{s}_1 > \hat{s}_2) = \begin{cases} 1, & \text{if } \hat{s}_1 > \hat{s}_2 \\ 1/2, & \text{if } \hat{s}_1 = \hat{s}_2 \\ 0, & \text{if } \hat{s}_1 < \hat{s}_2 \end{cases} \tag{5}$$

Obviously, the possibility degree $p(\hat{s}_1 \geq \hat{s}_2)$ satisfies the following properties:

1) $0 \leq p(\hat{s}_1 \geq \hat{s}_2) \leq 1$;

2) $p(\hat{s}_1 \geq \hat{s}_2) + p(\hat{s}_2 \geq \hat{s}_1) = 1$. Especially, $p(\hat{s}_1 \geq \hat{s}_1) = p(\hat{s}_2 \geq \hat{s}_2) = \frac{1}{2}$.

3 Some Aggregation Operators

In the following, we develop some operators for aggregating triangular fuzzy linguistic variables.

Definition 3.1. *Let* $FLA: \hat{S}^n \to \hat{S}$, *if*

$$FLA(\hat{s}_1, \hat{s}_2, ..., \hat{s}_n) = \frac{1}{n}(\hat{s}_1 \oplus \hat{s}_2 \oplus \cdots \oplus \hat{s}_n) \tag{6}$$

where $\hat{s}_i \in \hat{S}$, $i = 1,2,...,n$, *then* FLA *is called a fuzzy linguistic averaging (FLA) operator.*

Definition 3.2. *Let* $FLWA: \hat{S}^n \to \hat{S}$, *if*

$$FLWA_w(\hat{s}_1, \hat{s}_2, ..., \hat{s}_n) = w_1 \hat{s}_1 \oplus w_2 \hat{s}_2 \oplus \cdots \oplus w_n \hat{s}_n \tag{7}$$

where $w = (w_1, w_2, ..., w_n)^T$ *is the weighting vector of the* $\hat{s}_i$, $\hat{s}_i \in \hat{S}$, $w_i \geq 0$, $i = 1,2,...,n$, $\sum_{i=1}^{n} w_i = 1$ *then* $FLWA$ *is called a triangular fuzzy linguistic weighted averaging (FLWA) operator.*

In [15], Yager introduced the ordered weighted averaging (OWA) operator, whose fundamental feature is the reordering step. The OWA operator combines the arguments by giving weights to the values in relation to their ordering position, and diminishes the importance of extreme values by increasing the importance of central ones. In the following, we develop a fuzzy linguistic OWA operator to accommodate the situations where the input arguments are triangular fuzzy linguistic variables.

Definition 3.3. *A fuzzy linguistic ordered weighted geometric (FLOWA) operator of dimension* n *is a mapping* $FLOWA: \hat{S}^n \to \hat{S}$ *that has associated with* it a weighting *vector* $\omega = (\omega_1, \omega_2, ..., \omega_n)^T$ *such that* $\omega_j \geq 0$, $j = 1,2,...,n$, $\sum_{j=1}^{n} \omega_j = 1$. *Furthermore*

$$FLOWA_\omega(\hat{s}_1, \hat{s}_2, ..., \hat{s}_n) = \omega_1 \hat{r}_1 \oplus \omega_2 \hat{r}_2 \oplus \cdots \oplus \omega_n \hat{r}_n \tag{8}$$

where $\hat{r}_j$ *is the* j *th largest of* $\hat{s}_i$ $(i = 1, 2, ..., n)$, $\hat{s}_i \in \hat{S}$. *Especially, if* $\omega = (1/n, 1/n, ..., 1/n)^T$, *then the FLOWA operator is reduced to the FLA operator.*

To rank these triangular fuzzy linguistic variables $\hat{s}_j$ $(j = 1,2,..., n)$, we first compare each argument $\hat{s}_i$ with all triangular fuzzy linguistic variables $\hat{s}_j$

$(j = 1,2,...,n)$ by using (3), and let $p_{ij} = p(\hat{s}_i \geq \hat{s}_j)$. Then we construct a complementary matrix [16-22] $P = (p_{ij})_{n\times n}$, where

$$p_{ij} \geq 0,\ p_{ij} + p_{ji} = 1,\ p_{ii} = 0.5,\ \ i, j = 1,2,...,n \tag{9}$$

Summing all the elements in each line of matrix P, we have

$$p_i = \sum_{j=1}^{n} p_{ij},\ \ i = 1,2,...,n \tag{10}$$

Then we can rank the arguments $\hat{s}_i\ (i = 1,2,...,n)$ in descending order in accordance with the values of $p_i\ (i = 1,2,...,n)$.

Yager and Filev [23] introduced an induced ordered weighted averaging (IOWA) operator. The IOWA operator allows the introduction of semantics or meaning in the aggregation of arguments, and therefore allows for better control over the aggregation stage developed in the resolution process. Below we develop an induced FLOWA (IFLOWA) operator to accommodate the situations where the input arguments are triangular fuzzy linguistic variables.

Definition 3.4. *An IFLOWA operator is defined as:*

$$IFLOWA_{\omega}\left(\langle \delta_1, \hat{s}_1\rangle, \langle \delta_2, \hat{s}_2\rangle ..., \langle \delta_n, \hat{s}_n\rangle\right) = \omega_1 \hat{s}_{\gamma_1} \oplus \omega_2 \hat{s}_{\gamma_2} \oplus \cdots \oplus \omega_n \hat{s}_{\gamma_n} \tag{11}$$

where $\omega = (\omega_1, \omega_2, ..., \omega_n)^T$ *is a weighting vector,* $\omega_j \geq 0$, $j = 1,2,...,n$, $\sum_{j=1}^{n} \omega_j = 1$, $\hat{s}_{\gamma_j}$ *is the* $\hat{s}_i$ *value of the FLOWA pair* $\langle \delta_i, \hat{s}_i \rangle$ *having the* j *th largest* δ_i *(*$i = 1, 2, ..., n$*), and* δ_i *in* $\langle \delta_i, \hat{s}_i \rangle$ *is referred to as the order inducing variable and* $\hat{s}_i$ *as the triangular fuzzy linguistic argument variable. Especially, if* $\omega = (1/n, 1/n, ..., 1/n)^T$*, then IFLOWA is reduced to the FLA operator; if* $\delta_i = \hat{s}_i$*, for all* i*, then IFLOWA is reduced to the FLOWA operator; if* $\delta_i = No.i$*, for all* i*, where* $No.i$ *is the ordered position of* $\hat{s}_i$ *(*$i = 1, 2, ..., n$*), then IFLOWA is reduced to the FLWA operator.*

However, if there is a tie between $\langle \delta_i, \hat{s}_i \rangle$ and $\langle \delta_j, \hat{s}_j \rangle$ with respect to order inducing variables. In this case, we can follow the policy presented by Yager and Filev [23], that is, to replace the arguments of the tied objects by the average of the arguments of the tied objects, i.e., we replace the argument component of each of $\langle \delta_i, \hat{s}_i \rangle$ and $\langle \delta_j, \hat{s}_j \rangle$ by their average $(\hat{s}_i \oplus \hat{s}_j)/2$. If k items are tied, then we replace these by k replicas of their average.

The IFLOWA operator reflects the fuzzy majority by utilizing a fuzzy linguistic quantifier [14] to calculate its weighting vector. In the case of a non-decreasing proportional quantifier Q, the weighting vector can be obtained by the following expression:

$$\omega_k = Q\left(\frac{k}{n}\right) - Q\left(\frac{k-1}{n}\right), \quad k = 1,2,...,n \tag{12}$$

where

$$Q(r) = \begin{cases} 0 & if\ r < a, \\ \dfrac{r-a}{b-a} & if\ a \le r \le b, \\ 1 & if\ r > b \end{cases} \tag{13}$$

with $a, b, r \in [0, 1]$.

4 A Method for Group Decision Making with Triangular Fuzzy Linguistic Variables

Based on the above operators, we develop a practical method for ranking alternatives in group decision making with triangular fuzzy linguistic variables as follows.

Step 1. For a group decision making problem with fuzzy linguistic information. Let $X = \{x_1, x_2, ..., x_n\}$ be the set of alternatives, and $G = \{G_1, G_2, ..., G_m\}$ be the set of attributes. Let $w = (w_1, w_2, ..., w_m)^T$ be the weight vector of attributes, where $w_i \ge 0,\ i = 1,2,...,m,\ \sum_{i=1}^{m} w_i = 1$. Let $D = \{d_1, d_2, ..., d_m\}$ be the set of DMs, and $v = (v_1, v_2, ..., v_m)^T$ be the weight vector of DMs, where $v_l \ge 0,\ l = 1,2,...,m,$ $\sum_{l=1}^{m} v_l = 1$. Suppose that $\hat{A}^{(k)} = (\hat{a}_{ij}^{(k)})_{m\times n}$ is the fuzzy linguistic decision matrix, where $\hat{a}_{ij}^{(k)} \in \hat{S}$ is a triangular fuzzy linguistic variable, given by the DM $d_k \in D$, for the alternative $x_j \in X$ with respect to the attribute $G_i \in G$.

Step 2. Utilize the IFLOWA operator

$$\hat{a}_{ij} = IFLOWA_{\omega}\left(\left\langle v_1, \hat{a}_{ij}^{(1)}\right\rangle, \left\langle v_2, \hat{a}_{ij}^{(2)}\right\rangle, \cdots, \left\langle v_l, \hat{a}_{ij}^{(l)}\right\rangle\right), \quad i = 1,2,...,m;\ j = 1,2,...,n \tag{14}$$

to aggregate all the decision matrices $\hat{A}^{(k)}\ (k = 1,2,...,l)$ into a collective decision matrix $\hat{A} = (\hat{a}_{ij})_{m\times n}$, where $v = (v_1, v_2, ..., v_l)^T$ is the weight vector of DMs,

$\omega = (\omega_1, \omega_2, ..., \omega_l)^T$ is the weighting vector of the IFLOWA operator, $\omega_k \geq 0, \ k = 1,2,...,l, \sum_{k=1}^{l} \omega_k = 1$.

Step 3. Utilize the FLWA operator

$$\hat{a}_j = FLWA_w(\hat{a}_{1j}, \hat{a}_{2j}, ..., \hat{a}_{mj}) = w_1\hat{a}_{1j} \oplus w_2\hat{a}_{2j} \oplus \cdots \oplus w_m\hat{a}_{mj}, \quad j = 1,2,...,n \tag{15}$$

to derive the collective overall preference value $\hat{a}_j$ of the alternative x_j, where $w = (w_1, w_2, ..., w_m)^T$ is the weight vector of attributes.

Step 4. Compare each $\hat{a}_j$ with all $\hat{a}_i$ $(i = 1,2,...,n)$ by using (3), and develop a complementary matrix $P = (p_{ij})_{n \times n}$, where $p_{ij} = p(\hat{a}_i \geq \hat{a}_j)$.

Step 5. Rank $\hat{a}_j (j = 1,2,...,n)$ in descending order in accordance with the values of p_i $(i = 1,2,...,n)$ obtained by using (11).

Step 6. Rank all the alternatives x_j $(j = 1,2,...,n)$ and select the best one(s) in accordance with $\hat{a}_j$ $(j = 1,2,...,n)$.

Step 7. End.

5 Illustrative Example

In this section, a group decision making problem of evaluating university faculty for tenure and promotion (adapted from [24]) is used to illustrate the proposed procedure.

A practical use of the proposed approach involves the evaluation of university faculty for tenure and promotion. The criteria (attributes) used at some universities are G_1 : teaching, G_2 : research, and G_3 : service (whose weight vector $w = (0.36, 0.31, 0.33)^T$). Five faculty candidates (alternatives) x_j $(j = 1, 2, 3, 4, 5)$ are evaluated using the label set (listed in Section 2) by three DMs d_k $(k = 1,2,3)$ (whose weight vector $v = (0.4, 0.5, 0.1)^T$) under these three attributes, as listed in Tables 1-3.

Table 1. Fuzzy linguistic decision matrix $A^{(1)}$

G_i	x_1	x_2	x_3	x_4	x_5
G_1	(s_{-2}, s_{-1}, s_0)	(s_{-1}, s_0, s_1)	(s_{-2}, s_{-1}, s_1)	(s_1, s_3, s_4)	(s_0, s_1, s_3)
G_2	(s_0, s_2, s_3)	(s_1, s_3, s_4)	(s_{-3}, s_{-1}, s_0)	(s_{-1}, s_1, s_2)	(s_1, s_2, s_3)
G_3	(s_1, s_2, s_4)	(s_2, s_3, s_4)	(s_1, s_3, s_4)	(s_{-2}, s_{-1}, s_1)	(s_0, s_1, s_2)

Table 2. Fuzzy linguistic decision matrix $A^{(2)}$

G_i	x_1	x_2	x_3	x_4	x_5
G_1	(s_{-1}, s_0, s_1)	(s_1, s_2, s_4)	(s_1, s_2, s_3)	(s_1, s_2, s_4)	(s_{-1}, s_1, s_2)
G_2	(s_{-2}, s_{-1}, s_0)	(s_{-1}, s_0, s_2)	(s_1, s_2, s_3)	(s_0, s_1, s_2)	(s_2, s_3, s_4)
G_3	(s_1, s_2, s_3)	(s_2, s_3, s_4)	(s_0, s_1, s_3)	(s_0, s_1, s_2)	(s_{-1}, s_0, s_1)

Table 3. Fuzzy linguistic decision matrix $A^{(3)}$

G_i	x_1	x_2	x_3	x_4	x_5
G_1	(s_{-1}, s_0, s_2)	(s_{-1}, s_0, s_1)	(s_{-1}, s_1, s_2)	(s_2, s_3, s_4)	(s_0, s_1, s_3)
G_2	(s_1, s_3, s_4)	(s_{-2}, s_{-1}, s_0)	(s_0, s_2, s_3)	(s_{-1}, s_1, s_3)	(s_1, s_2, s_4)
G_3	(s_2, s_3, s_4)	(s_1, s_2, s_3)	(s_{-2}, s_0, s_1)	(s_2, s_3, s_4)	(s_{-1}, s_0, s_1)

In what follows, we utilize the method developed in this paper to get the most desirable alternative(s):

Step 1. Utilize (11) (by using the linguist quantifier "*most*", with the corresponding weighting vector $\omega=(1/15,10/15,4/15)^T$):

$$\hat{a}_{ij} = IFLOWA_{\omega}\left(\left\langle v_1, \hat{a}_{ij}^{(1)}\right\rangle, \left\langle v_2, \hat{a}_{ij}^{(2)}\right\rangle, \left\langle v_3, \hat{a}_{ij}^{(3)}\right\rangle\right),\ i=1,2,3;\ j=1,2,3,4,5$$

to aggregate all the decision matrices $\hat{A}^{(k)}\ (k=1,2,3)$ into a collective decision matrix $\hat{A}=(\hat{a}_{ij})_{3\times 4}$ (see Table 4).

Table 4. The collective Fuzzy linguistic decision matrix $\hat{A}$

G_i	x_1	x_2	x_3	x_4	x_5
G_1	$(s_{-1.67}, s_{-0.67}, s_{0.60})$	$(s_{-0.87}, s_{0.13}, s_{1.20})$	$(s_{-1.53}, s_{-0.27}, s_{1.40})$	$(s_{1.27}, s_{2.93}, s_{4.00})$	$(s_{-0.07}, s_{1.00}, s_{2.93})$
G_2	$(s_{0.13}, s_{2.07}, s_{3.07})$	$(s_{0.07}, s_{1.73}, s_{2.80})$	$(s_{-1.93}, s_{0.00}, s_{1.00})$	$(s_{-0.93}, s_{1.00}, s_{2.27})$	$(s_{1.07}, s_{2.07}, s_{3.33})$
G_3	$(s_{1.27}, s_{2.27}, s_{3.93})$	$(s_{1.73}, s_{2.73}, s_{3.73})$	$(s_{0.13}, s_{2.07}, s_{3.13})$	$(s_{-0.80}, s_{0.20}, s_{1.87})$	$(s_{-0.38}, s_{0.67}, s_{1.67})$

Step 2. Utilize the FLWA operator

$$\hat{a}_j = FLWA_w(\hat{a}_{1j}, \hat{a}_{2j}, \hat{a}_{3j}),\ j=1,2,3,4,5$$

to aggregate $\hat{a}_{ij}\ (i=1,2,3)$ corresponding to the alternative x_j, and then get the collective overall preference value $\hat{a}_j$ of the alternative x_j:

$$\hat{a}_1 = (s_{-0.142}, s_{1.150}, s_{2.465}),\ \hat{a}_2 = (s_{0.279}, s_{1.484}, s_{2.531}),\ \hat{a}_3 = (s_{-1.106}, s_{0.586}, s_{1.847})$$

$$\hat{a}_4 = (s_{-0.095}, s_{1.431}, s_{2.761}),\ \hat{a}_5 = (s_{0.181}, s_{1.223}, s_{2.638})$$

Step 3. Suppose that the DM's attitude is neutral to the risk, i.e., $\rho = 0.5$, then compare each $\hat{a}_i$ with all the $\hat{a}_j$ $(j = 1,2,3,4,5)$ by using (3), and develop a complementary matrix:

$$P = \begin{bmatrix} 0.5 & 0.3821 & 0.7427 & 0.4163 & 0.4351 \\ 0.6179 & 0.5 & 0.8683 & 0.5205 & 0.5556 \\ 0.2573 & 0.1317 & 0.5 & 0.1861 & 0.1906 \\ 0.5837 & 0.4795 & 0.8139 & 0.5 & 0.5235 \\ 0.5649 & 0.4444 & 0.8094 & 0.4765 & 0.5 \end{bmatrix}$$

Summing all elements in each line of matrix P, we have

$$p_1 = 2.4762,\ p_2 = 3.0623,\ p_3 = 1.2657,\ p_4 = 2.9006,\ p_5 = 2.7952$$

Then we rank $\hat{a}_j$ $(j = 1,2,3,4,5)$ in descending order in accordance with the values of p_j $(j = 1,2,3,4,5)$): $\hat{a}_2 > \hat{a}_4 > \hat{a}_5 > \hat{a}_1 > \hat{a}_3$.

Step 4. Rank all the alternatives x_j $(j = 1,2,3,4,5)$ and select the best one(s) in accordance with $\hat{a}_j$ $(j = 1,2,3,4,5)$, then we get $x_2 \succ x_4 \succ x_5 \succ x_1 \succ x_3$, and thus, the most desirable alternative is x_2.

6 Concluding Remarks

In group decision making with linguistic information, the DMs sometimes provide only fuzzy linguistic information because of time pressure, lack of knowledge, and their limited attention and information processing capabilities. In this paper, we have investigated the group decision making problem with triangular fuzzy linguistic variables. We have defined the concepts of triangular fuzzy linguistic variable and some of its operational laws, and then proposed some operators for aggregating triangular fuzzy linguistic variables. Based on these aggregation operators, we have developed a practical method for group decision making with triangular fuzzy linguistic variables. In future research, our work will focus on the application of triangular fuzzy linguistic variables in the field of computing with words.

Acknowledgements

The work was supported by the National Natural Science Foundation of China (No.70571087and No.70321001), China Postdoctoral Science Foundation (No. 20060390051), and the National Science Fund for Distinguished Young Scholars of China (No.70625005).

References

1. Herrera, F., Herrera-Viedma, E., Verdegay, J.L.: A sequential selection process in group decision making with linguistic assessment. Information Sciences 85, 223–239 (1995)
2. Herrera, F., Herrere-Viedma, E., Verdegay, J.L.: Direct approach processes in group decision making using linguistic OWA operators. Fuzzy Sets and Systems 79, 175–190 (1996)

3. Herrera, F., Herrera-Viedma, E., Verdegay, J.L.: A model of consensus in group decision making under linguistic assessments. Fuzzy Sets and Systems 78, 73–87 (1996)
4. Herrera, F., Herrera-Viedma, E., Verdegay, J.L.: A rational consensus model in group decision making using linguistic assessments. Fuzzy Sets and Systems 88, 31–49 (1997)
5. Herrera, F., Herrera-Viedma, E.: Linguistic decision analysis: steps for solving decision problems under linguistic information. Fuzzy Sets and Systems 115, 67–82 (2000)
6. Herrera, F., Martínez, L.: A model based on linguistic 2-tuples for dealing with multigranular hierarchical linguistic contexts in multi-expert decision making. IEEE Transactions on Systems, Man, and Cybernetics-Part B 31, 227–234 (2001)
7. Bordogna, G., Fedrizzi, M., Pasi, G.: A linguistic modeling of consensus in group decision making based on OWA operators. IEEE Transactions on Systems, Man, and Cybernetics-Part A 27, 126–132 (1997)
8. Marimin, M., Umano, M., Hatono, I., Tamura, H.: Linguistic labels for expressing fuzzy preference relations in fuzzy group decision making. IEEE Transactions on Systems, Man, and Cybernetics-Part B 28, 205–218 (1998)
9. Xu, Z.S.: A method based on linguistic assessments and GIOWA operator for multiple attribute group decision making. Journal of Systems Science and mathematics 24, 225–231 (2004)
10. Xu, Z.S.: A method based on linguistic aggregation operators for group decision making with linguistic preference relations. Information Sciences 166, 19–30 (2004)
11. Xu, Z.S.: Uncertain Multiple Attribute Decision Making: Methods and Applications. Tsinghua University Press, Beijing (2004)
12. Xu, Z.S.: Deviation measures of linguistic preference relations in group decision making. Omega 33, 249–254 (2005)
13. Xu, Z.S.: A note on linguistic hybrid arithmetic averaging operator in group decision making with linguistic information. Group Decision and Negotiation 15, 581–591 (2006)
14. Xu, Z.S.: Integrating multiple types of incomplete linguistic preference relations in multiperson decision making. In: Wang, L., Jiao, L., Shi, G., Li, X., Liu, J. (eds.) FSKD 2006. LNCS (LNAI), vol. 4223, pp. 300–309. Springer, Heidelberg (2006)
15. Yager, R.R.: On ordered weighted averaging aggregation operators in multicriteria decisionmaking. IEEE Transactions on Systems, Man, and Cybernetics 18, 183–190 (1988)
16. Xu, Z.S.: Study on the relation between two classes of scales in AHP. Systems Engineering—Theory & Practice 19(7), 97–101 (1999)
17. Chiclana, F., Herrera, F., Herrera-Viedma, E.: Integrating multiplicative preference relations in a multipurpose decision making model based on fuzzy preference relations. Fuzzy Sets and Systems 112, 277–291 (2001)
18. Xu, Z.S., Da, Q.L.: The uncertain OWA operator. International Journal of Intelligent Systems 17, 569–575 (2002)
19. Xu, Z.S.: A survey of preference relations. International Journal of General Systems 36, 179–203 (2007)
20. Xu, Z.S., Da, Q.L.: An approach to improving consistency of fuzzy preference matrix. Fuzzy Optimization and Decision Making 2, 3–12 (2003)
21. Xu, Z.S.: Goal programming models for obtaining the priority vector of incomplete fuzzy preference relation. International Journal of Approximate Reasoning 36, 261–270 (2004)
22. Xu, Z.S., Da, Q.L.: A least deviation method to obtain a priority vector of a fuzzy preference relation. European Journal of Operational Research 164, 206–216 (2005)
23. Yager, R.R., Filev, D.P.: Induced ordered weighted averaging operators. IEEE Transactions on Systems, Man, and Cybernetics-Part B 29, 141–150 (1999)
24. Bryson, N., Mobolurin, A.: An action learning evaluation procedure for multiple criteria decision making problems. European Journal of Operational Research 96, 379–386 (1995)

Sparse Kernel Modelling: A Unified Approach

S. Chen[1], X. Hong[2], and C.J. Harris[1]

[1] School of Electronics and Computer Sciences, University of Southampton, Southampton SO17 1BJ, U.K.
{sqc,cjh}@ecs.soton.ac.uk
[2] School of Systems Engineering, University of Reading, Reading RG6 6AY, U.K.
x.hong@reading.ac.uk

Abstract. A unified approach is proposed for sparse kernel data modelling that includes regression and classification as well as probability density function estimation. The orthogonal-least-squares forward selection method based on the leave-one-out test criteria is presented within this unified data-modelling framework to construct sparse kernel models that generalise well. Examples from regression, classification and density estimation applications are used to illustrate the effectiveness of this generic sparse kernel data modelling approach.

1 Introduction

The objective of modelling from data is not that the model simply fits the training data well. Rather, the goodness of a model is characterised by its generalisation capability, interpretability and ease for knowledge extraction. All these desired properties depend crucially on the ability to construct appropriate sparse models by the modelling process, and a basic principle in practical data modelling is the parsimonious principle of ensuring the smallest possible model that explains the training data. Recently considerable research efforts have been focused on sparse kernel data modelling techniques [1,2,3,4,5,6]. Sparse kernel modelling methods typically use every training input data as a kernel. A sparse representation is then sought based on various criteria by making as many kernel weights to (near) zero values as possible. A different approach to these sparse kernel modelling methods is the forward selection using the orthogonal least squares (OLS) algorithm [7,8], developed in the late 80s for nonlinear system modelling, which remains highly popular for data modelling practicians.

Since its derivation, many enhanced variants of the OLS forward-selection algorithm have been proposed by incorporating the new developments from machining learning and the approach has extended its application to all the areas of data modelling, including regression, classification and kernel density estimation [9,10,11,12,13,14,15]. This contribution continues this theme, and it presents a unified framework for sparse kernel modelling that include all the three classes of data modelling applications, namely, regression, classification and probability density function (PDF) estimation. Based on this unified data-modelling framework, the OLS forward selection algorithm using the leave-one-out (LOO) test

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 27–36, 2007.

criteria and local regularisation (LR) is employed to construct sparse kernel models with excellent generalisation capability. Experimental results are included to demonstrate the effectiveness of the OLS forward selection algorithm based on the LOO test criteria within the proposed unified data-modelling framework.

2 A Unified Framework for Data Modelling

Given the training data set, $D_N = \{\mathbf{x}_k, y_k\}_{k=1}^N$, where $\mathbf{x}_k = [x_{1,k}\ x_{2,k} \cdots x_{m,k}]^T \in \mathcal{R}^m$ is an observation sample and y_k is the target or desired response for $\mathbf{x}_k$, the task is to infer a kernel model of the form

$$\hat{y} = \hat{f}(\mathbf{x}; \boldsymbol{\beta}_N, \rho) = \sum_{i=1}^{N} \beta_i K_\rho(\mathbf{x}, \mathbf{x}_i) \tag{1}$$

to capture the underlying data generating mechanism, where $\hat{y}$ denotes the model output, $\boldsymbol{\beta}_N = [\beta_1\ \beta_2 \cdots \beta_N]^T$ is the kernel weight vector and $K_\rho(\bullet, \bullet)$ is the chosen kernel function with a kernel width ρ. Many types of kernel function can be employed and a commonly used one is the Gaussian function of the form

$$K_\rho(\mathbf{x}, \mathbf{c}_k) = \frac{1}{(2\pi\rho^2)^{m/2}} e^{-\frac{\|\mathbf{x}-\mathbf{c}_k\|^2}{2\rho^2}}, \tag{2}$$

where $\mathbf{c}_k \in \mathcal{R}^m$ is the k-th kernel centre vector. For regression and classification problems, the factor $\frac{1}{(2\pi\rho^2)^{m/2}}$ can be combined into kernel weights β_i. The generic kernel model (1) is defined by placing a kernel at each of the training input samples $\mathbf{x}_k$ and forming a linear combination of all the bases defined on the training data set. A sparse representation is then sought by selecting only N_s significant regressors from the full regressor set, where $N_s \ll N$.

The underlying data generating mechanism is governed by $y = f(\mathbf{x}) + \varepsilon$, where ε is a white process representing the observation noise. For regression problems, the unknown mapping $f : \mathcal{R}^m \to \mathcal{R}$. Regression is a supervised learning problem, as the desired response $y_k \in \mathcal{R}$ for the training data point $\mathbf{x}_k$ is given. For two-class classification problems, the unknown mapping $f : \mathcal{R}^m \to \{-1, +1\}$. The estimated class label for the pattern vector $\mathbf{x}_k$ is given by $\tilde{y}_k = \text{sgn}(\hat{y}_k)$ with

$$\text{sgn}(y) = \begin{cases} -1, & y \le 0, \\ +1, & y > 0. \end{cases} \tag{3}$$

Classification is also a supervised learning problem, since the correct label $y_k \in \{-1, +1\}$ for the training data point $\mathbf{x}_k$ is provided. For PDF estimation problems, the data $\{\mathbf{x}_k\}_{k=1}^N$ are drawn from a unknown density $f : \mathcal{R}^m \to \mathcal{R}_+$. Because f is a PDF, $f(\mathbf{x}) \ge 0$ for $\mathbf{x} \in \mathcal{R}^m$ and $\int_{\mathcal{R}^m} f(\mathbf{u})\, d\mathbf{u} = 1$. Thus, a kernel in a kernel density estimate must satisfy $K_\rho(\mathbf{x}, \mathbf{c}_k) \ge 0$ and $\int_{\mathcal{R}^m} K_\rho(\mathbf{u}, \mathbf{c}_k)\, d\mathbf{u} = 1$. Moreover the kernel weights must satisfy the nonnegative constraint

$$\beta_k \ge 0,\ 1 \le k \le N, \tag{4}$$

and the unity constraint

$$\boldsymbol{\beta}_N^T \mathbf{1}_N = 1, \tag{5}$$

where $\mathbf{1}_N$ denotes the vector of ones with dimension N. Kernel density estimation is an unsupervised learning problem because the desired response is unknown for each training data point $\mathbf{x}_k$. This difficult is circumvented by "inventing" a target function y_k for $\mathbf{x}_k$, so that the problem becomes a constrained regression one with the constraints (4) and (5). In particular, we choose y_k to be the value of the Parzen window estimate [16,17] at point $\mathbf{x}_k$. This choice of the desired response for density estimation is fully justified in [13].

Let the modelling error at training data point $\mathbf{x}_k$ be $\epsilon_k = y_k - \hat{y}_k$, where

$$\hat{y}_k = [K_\rho(\mathbf{x}_k, \mathbf{x}_1)\ K_\rho(\mathbf{x}_k, \mathbf{x}_2) \cdots K_\rho(\mathbf{x}_k, \mathbf{x}_N)]\ \boldsymbol{\beta}_N = \boldsymbol{\phi}^T(k)\boldsymbol{\beta}_N. \tag{6}$$

Define $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1\ \boldsymbol{\phi}_2 \cdots \boldsymbol{\phi}_N]$ with $\boldsymbol{\phi}_k = [K_\rho(\mathbf{x}_1, \mathbf{x}_k)\ K_\rho(\mathbf{x}_2, \mathbf{x}_k) \cdots K_\rho(\mathbf{x}_N, \mathbf{x}_k)]^T$ for $1 \le k \le N$, $\mathbf{y} = [y_1\ y_2 \cdots y_N]^T$ and $\boldsymbol{\epsilon} = [\epsilon_1\ \epsilon_2 \cdots \epsilon_N]^T$. The regression model (1) over the training data set D_N can then be expressed in the matrix form

$$\mathbf{y} = \boldsymbol{\Phi}\,\boldsymbol{\beta}_N + \boldsymbol{\epsilon}. \tag{7}$$

Let an orthogonal decomposition of the regression matrix $\boldsymbol{\Phi}$ be $\boldsymbol{\Phi} = \mathbf{W}\,\mathbf{A}_N$, where $\mathbf{A}_N$ is the $N \times N$ upper triangular matrix with unity diagonal elements, and $\mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2 \cdots \mathbf{w}_N]$ with orthogonal columns satisfying $\mathbf{w}_i^T \mathbf{w}_j = 0$, if $i \ne j$. The regression model (7) can alternatively be expressed as

$$\mathbf{y} = \mathbf{W}\,\mathbf{g}_N + \boldsymbol{\epsilon}, \tag{8}$$

where the weight vector $\mathbf{g}_N = [g_1\ g_2 \cdots g_N]^T$ satisfies the triangular system $\mathbf{A}_N \boldsymbol{\beta}_N = \mathbf{g}_N$. The model (6) is equivalently expressed by

$$\hat{y}_k = \mathbf{w}^T(k)\,\mathbf{g}_N, \tag{9}$$

where $\mathbf{w}^T(k) = [w_{k,1}\ w_{k,2} \cdots w_{k,N}]$ is the k-th row of $\mathbf{W}$.

3 Orthogonal-Least-Squares Algorithm

As established in the previous section, the regression, classification and PDF estimation can all be unified within the common regression modelling framework. Therefore, the OLS forward selection based on the LOO test criteria and local regularisation (OLS-LOO-LR) [10] provides an efficient algorithm to construct a sparse kernel model that generalise well.

3.1 Sparse Kernel Regression Model Construction

The LR aided least squares solution for the weight parameter vector $\mathbf{g}_N$ can be obtained by minimising the following regularised error criterion [11]

$$J_R(\mathbf{g}_N, \boldsymbol{\lambda}) = \boldsymbol{\epsilon}^T \boldsymbol{\epsilon} + \mathbf{g}_N^T \boldsymbol{\Lambda} \mathbf{g}_N, \tag{10}$$

where $\boldsymbol{\lambda} = [\lambda_1 \ \lambda_2 \cdots \lambda_N]^T$ is the vector of regularisation parameters, and $\boldsymbol{\Lambda} = \text{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_N\}$. Applying the evidence procedure results in the following iterative updating formulas for the regularisation parameters [9]

$$\lambda_i^{\text{new}} = \frac{\gamma_i^{\text{old}}}{N - \gamma^{\text{old}}} \frac{\boldsymbol{\epsilon}^T \boldsymbol{\epsilon}}{g_i^2}, \ 1 \le i \le N, \tag{11}$$

where g_i for $1 \le i \le N$ denote the current estimated parameter values, and

$$\gamma = \sum_{i=1}^{N} \gamma_i \quad \text{with} \quad \gamma_i = \frac{\mathbf{w}_i^T \mathbf{w}_i}{\lambda_i + \mathbf{w}_i^T \mathbf{w}_i}. \tag{12}$$

Typically a few iterations (less than 10) are sufficient to find a (near) optimal $\boldsymbol{\lambda}$. The use of LR is known to be capable of providing very sparse solutions [2,11].

For regression, the OLS-LOO-LR algorithm selects a sparse model by incrementally minimising the LOO mean square error (MSE) criterion, which is a measure of the model's generalisation performance [10,14,18]. At the n-th stage of the OLS selection procedure, an n-term model is selected. The LOO test error, denoted as $\epsilon_k^{(n,-k)}$, for the selected n-term model is defined as [10,14]

$$\epsilon_k^{(n,-k)} = \epsilon_k^{(n)} / \eta_k^{(n)}, \tag{13}$$

where $\epsilon_k^{(n)}$ is the usual n-term modelling error and $\eta_k^{(n)}$ is the associated LOO error weighting. The LOO MSE for the model with a size n is then defined by

$$J_n = \frac{1}{N} \sum_{k=1}^{N} \left(\epsilon_k^{(n,-k)} \right)^2 = \frac{1}{N} \sum_{k=1}^{N} \left(\epsilon_k^{(n)} \right)^2 / \left(\eta_k^{(n)} \right)^2. \tag{14}$$

This LOO MSE can be computed efficiently due to the fact that $\epsilon_k^{(n)}$ and $\eta_k^{(n)}$ can be calculated recursively according to [10,14]

$$\epsilon_k^{(n)} = \epsilon_k^{(n-1)} - w_{k,n} g_n \tag{15}$$

and

$$\eta_k^{(n)} = \eta_k^{(n-1)} - w_{k,n}^2 / \left(\mathbf{w}_n^T \mathbf{w}_n + \lambda_n \right), \tag{16}$$

respectively, where $w_{k,n}$ is the k-th element of $\mathbf{w}_n$. The selection is carried out as follows. At the n-th stage of the selection procedure, a model term is selected among the remaining n to N candidates if the resulting n-term model produces the smallest LOO MSE J_n. The selection procedure is terminated when

$$J_{N_s+1} \ge J_{N_s}, \tag{17}$$

yielding an N_s-term sparse model. The LOO statistic J_n is at least locally convex with respect to the model size n [14]. Thus, there exists an "optimal" model size N_s such that for $n \le N_s$ J_n decreases as n increases while the condition (17) holds. The sparse regression model selection procedure is now summarised.

: Set $\lambda_i = 10^{-6}$ for $1 \le i \le N$, and set iteration index $I = 1$.
: Given the current $\boldsymbol{\lambda}$ and with the following initial conditions

$$\epsilon_k^{(0)} = y_k, \eta_k^{(0)} = 1,\ 1 \le k \le N,\ \text{and}\ J_0 = \mathbf{y}^T\mathbf{y}/N, \tag{18}$$

use the OLS-LOO procedure [10] to select a subset model with N_I terms.
: Update $\boldsymbol{\lambda}$ using (11) and (12) with $N = N_I$. If the maximum iteration number (e.g. 10) is reached, stop; otherwise set $I+ = 1$ and go to .

3.2 Sparse Kernel Classifier Construction

The same LOO cross validation concept [18] is adopted to provide a measure of classifier's generalisation capability. Denote the test output of the LOO n-term model evaluated at the k-th data sample of D_N not used in training as $\hat{y}_k^{(n,-k)}$. The associated LOO signed decision variable is defined by

$$s_k^{(n,-k)} = \text{sgn}(y_k)\hat{y}_k^{(n,-k)} = y_k\hat{y}_k^{(n,-k)}, \tag{19}$$

where $\text{sgn}(y_k) = y_k$ since the class label $y_k \in \{-1, +1\}$. The LOO misclassification rate can be computed by

$$J_n = \frac{1}{N}\sum_{k=1}^{N}\mathcal{I}_d\left(s_k^{(n,-k)}\right), \tag{20}$$

where the indication function is defined by $\mathcal{I}_d(y) = 1$ if $y \le 0$ and $\mathcal{I}_d(y) = 0$ if $y > 0$. The LOO misclassification rate J_n can be evaluated efficiently because $s_k^{(n,-k)}$ can be calculated very fast [15]. Specifically, express the LOO signed decision variable as $s_k^{(n,-k)} = \psi_k^{(n)}/\eta_k^{(n)}$. The recursive formula for $\eta_k^{(n)}$ is given in (16), while $\psi_k^{(n)}$ can be represented using the recursive formula [15]

$$\psi_k^{(n)} = \psi_k^{(n-1)} + y_k g_n w_{k,n} - w_{k,n}^2/\left(\mathbf{w}_n^T\mathbf{w}_n + \lambda_n\right). \tag{21}$$

The OLS-LOO-LR algorithm described in Subsection 3.1 can readily be applied to select a sparse kernel classifier with some minor modifications. Moreover, extensive empirical experience has suggested that all the regularisation parameters λ_i, $1 \le i \le N$, can be set to a small positive constant λ, and there is no need to update them using the evidence procedure. The sparse kernel classifier selection procedure based on this OLS-LOO algorithm is now summarised.

Setting λ to a small positive number, and with the following initial conditions

$$\psi_k^{(0)} = 0 \quad \text{and} \quad \eta_k^{(0)} = 1 \quad \text{for} \quad 1 \le k \le N, \quad \text{and} \quad J_0 = 1, \tag{22}$$

use the OLS-LOO procedure [15] to select a subset model with N_s terms.

The LOO misclassification rate J_n is also locally convex with respect to the classifier's size n. Thus there exists an optimal model size N_s such that for $n \le N_s$ J_n decreases as n increases, while $J_{N_s} \le J_{N_s+1}$. Therefore the selection procedure is automatically terminated with a subset classifier containing only N_s significant kernels.

3.3 Sparse Kernel Density Estimator Construction

Since the kernel density estimation problem can be expressed as a constrained regression modelling, the OLS-LOO-LR algorithm detailed in Subsection 3.1 can be used to select a sparse kernel density estimate. After the structure determination using the OLS-LOO-LR algorithm, a sparse N_s-term subset kernel model is obtained. Let $\mathbf{A}_{N_s}$ denote the subset matrix of $\mathbf{A}_N$, corresponding to the selected N_s-term subset model. The kernel weight vector $\boldsymbol{\beta}_{N_s}$, computed from $\mathbf{A}_{N_s}\boldsymbol{\beta}_{N_s} = \mathbf{g}_{N_s}$, may not satisfy the constraints (4) and (5). However, we can recalculate $\boldsymbol{\beta}_{N_s}$ using the multiplicative nonnegative quadratic programming (MNQP) algorithm [3,6]. Since N_s is very small, the extra computation involved is small. Formally, this task is defined as follows. Find $\boldsymbol{\beta}_{N_s}$ for the model

$$\mathbf{y} = \boldsymbol{\Phi}_{N_s}\boldsymbol{\beta}_{N_s} + \boldsymbol{\epsilon}, \tag{23}$$

subject to the constraints

$$\beta_i \geq 0,\ 1 \leq i \leq N_s, \tag{24}$$

$$\boldsymbol{\beta}_{N_s}^T \mathbf{1}_{N_s} = 1, \tag{25}$$

where $\boldsymbol{\Phi}_{N_s}$ denotes the selected subset regression matrix and $\boldsymbol{\beta}_{N_s}^T = [\beta_1\ \beta_2 \cdots \beta_{N_s}]$. The kernel weight vector can be obtained by solving the following constrained nonnegative quadratic programming

$$\begin{aligned} &\min_{\boldsymbol{\beta}_{N_s}} \{\tfrac{1}{2}\boldsymbol{\beta}_{N_s}^T \mathbf{C}_{N_s}\boldsymbol{\beta}_{N_s} - \mathbf{v}_{N_s}^T\boldsymbol{\beta}_{N_s}\} \\ &\text{s.t. } \boldsymbol{\beta}_{N_s}^T \mathbf{1}_{N_s} = 1 \text{ and } \beta_i \geq 0,\ 1 \leq i \leq N_s, \end{aligned} \tag{26}$$

where $\mathbf{C}_{N_s} = \boldsymbol{\Phi}_{N_s}^T\boldsymbol{\Phi}_{N_s} = [c_{i,j}] \in \mathcal{R}^{N_s \times N_s}$ is the related design matrix and $\mathbf{v}_{N_s} = \boldsymbol{\Phi}_{N_s}^T\mathbf{y} = [v_1\ v_2 \cdots v_{N_s}]^T$. Although there exists no closed-form solution for this optimisation problem, the solution can readily be obtained iteratively using a modified version of the MNQP algorithm [3].

Specifically, the iterative updating equations for $\boldsymbol{\beta}_{N_s}$ are given by [6,13]

$$r_i^{<t>} = \beta_i^{<t>}\left(\sum_{j=1}^{N_s} c_{i,j}\beta_j^{<t>}\right)^{-1},\ 1 \leq i \leq N_s, \tag{27}$$

$$h^{<t>} = \left(\sum_{i=1}^{N_s} r_i^{<t>}\right)^{-1}\left(1 - \sum_{i=1}^{N_s} r_i^{<t>} v_i\right), \tag{28}$$

$$\beta_i^{<t+1>} = r_i^{<t>}\left(v_i + h^{<t>}\right), \tag{29}$$

where the superindex $^{<t>}$ denotes the iteration index and h is the Lagrangian multiplier. During the iterative procedure, some of the kernel weights may be driven to (near) zero [3,6]. The corresponding kernels can then be removed from the kernel model, leading to a further reduction in the subset model size.

Table 1. Comparison of modelling accuracy for the Boston housing data set. The results were averaged over 100 realizations and quoted as the mean±standard deviation.

algorithm	model size	training MSE	test MSE
OLS-LOO-LR	58.6 ± 11.3	12.9690 ± 2.6628	17.4157 ± 4.6670
SVM	243.2 ± 5.3	6.7986 ± 0.4444	23.1750 ± 9.0459

4 Empirical Data Modelling Results

Boston Housing Data Set. This was a regression benchmark data set, available at the UCI repository [19]. The data set comprised 506 data points with 14 variables. The task was to predict the median house value from the remaining 13 attributes. From the data set, 456 data points were randomly selected for training and the remaining 50 data points were used to form the test set. Because a Gaussian kernel was placed at each training data sample, there were $N = 456$ candidate regressors in the full regression model (1). The kernel width for the OLS-LOO-LR algorithm was determined via a grid-search based cross validation. The support vector machine (SVM) algorithm with the ε-insensitive cost function was also used to construct the regression model for this data set, as a comparsion. The three learning parameters of the SVM algorithm, the kernel width, error-band and trade-off parameters, were tuned via cross validation. Average results were given over 100 repetitions, and the two sparse Gaussian kernel models obtained by the OLS-LOO-LR and SVM algorithms, respectively, are compared in Table 1.

For the particular computational platform used in the experiment, the recorded average run time for the OLS-LOO-LR algorithm when the kernel width was fixed was 200 times faster than the SVM algorithm when the kernel width, error-band and trade-off parameters were chosen. It can be seen from Table 1 that the OLS-LOO-LR algorithm achieved better modelling accuracy with a much sparser model than the SVM algorithm. The test MSE of the SVM algorithm was poor. This was probably because the three learning paremeters, namely the kernel width, error-band and trade-off parameters, were not tuned to the optimal values. For this regression problem of input dimension 13 and data size $N \approx 500$, the grid search required by the SVM algorithm to tune the three learning parameters was expensive and the optimal values of the three learning parameters were hard to find.

Diabetes data. This two-class classification benchmark data set was originated in the UCI repository [19] and the data set used in the experiment was obtained from [20]. The feature space dimension was $m = 8$. There were 100 realisations of the data set, each having 468 training patterns and 300 test patterns. Seven existing state-of-the-art radial basis function (RBF) and kernel classifiers were compared in [20,21]. The results given in [20] were reproduced in Table 2. For the first 5 methods studied in [20], the nonlinear RBF network with 15 optimised Gaussian units was used. For the SVM algorithm with Gaussian kernel,

Table 2. Average classification test error rate in % over the 100 realizations of the diabetes data set. The first 7 results were quoted from [20].

algorithm	test error rate	model size
RBF-Network	24.29 ± 1.88	15
AdaBoost RBF-Network	26.47 ± 2.29	15
LP-Reg-AdaBoost	24.11 ± 1.90	15
QP-Reg-AdaBoost	25.39 ± 2.20	15
AdaBoost-Reg	23.79 ± 1.80	15
SVM	23.53 ± 1.73	not available
Kernel Fisher Discriminant	23.21 ± 1.63	468
OLS-LOO	23.00 ± 1.70	6.0 ± 1.0

no average model size was given in [20] but it could safely be assumed that it was much larger than 40. The kernel Fisher discriminant was the non-sparse optimal classifier using all the $N = 468$ training data samples as kernels.

The OLS-LOO algorithm was applied to construct sparse Gaussian kernel classifiers for this data set, and the results averaged over the 100 realisations are also listed in Table 2. It can be seen that the proposed OLS-LOO method compared favourably with the existing benchmark RBF and kernel classifier construction algorithms, both in terms of classification accuracy and model size.

Six-dimensional density estimation. The underlying density to be estimated was given by

$$f(\mathbf{x}) = \frac{1}{3}\sum_{i=1}^{3} \frac{1}{(2\pi)^{6/2}} \frac{1}{\det^{1/2}|\boldsymbol{\Gamma}_i|} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Gamma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)} \tag{30}$$

with

$$\begin{aligned} \boldsymbol{\mu}_1 &= [1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0]^T, \\ \boldsymbol{\Gamma}_1 &= \mathrm{diag}\{1.0, 2.0, 1.0, 2.0, 1.0, 2.0\}, \end{aligned} \tag{31}$$

$$\begin{aligned} \boldsymbol{\mu}_2 &= [-1.0\ -1.0\ -1.0\ -1.0\ -1.0\ -1.0]^T, \\ \boldsymbol{\Gamma}_2 &= \mathrm{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}, \end{aligned} \tag{32}$$

$$\begin{aligned} \boldsymbol{\mu}_3 &= [0.0\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0]^T, \\ \boldsymbol{\Gamma}_3 &= \mathrm{diag}\{2.0, 1.0, 2.0, 1.0, 2.0, 1.0\}. \end{aligned} \tag{33}$$

A training data set of $N = 600$ randomly drawn samples was used to construct kernel density estimates, and a separate test data set of $N_{\text{test}} = 10,000$ samples was used to calculate the L_1 test error for the resulting estimate according to

$$L_1 = \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} \left| f(\mathbf{x}_k) - \hat{f}(\mathbf{x}_k; \boldsymbol{\beta}_N, \rho) \right|. \tag{34}$$

The experiment was repeated $N_{\text{run}} = 100$ different random runs.

Table 3. Performance comparison for the six-dimensional three-Gaussian mixture

method	L_1 test error	kernel number
Parzen window estimate	$(3.5195 \pm 0.1616) \times 10^{-5}$	600 ± 0
SKD estimate of [12]	$(4.4781 \pm 1.2292) \times 10^{-5}$	14.9 ± 2.1
OLS-LOO-LR/MNQP	$(3.1134 \pm 0.5335) \times 10^{-5}$	9.4 ± 1.9

Simulation was used to test the proposed combined OLS-LOO-LR and MNQP algorithm and to compare its performance with the Parzen window estimator as well as our previous sparse kernel density (SKD) estimation algorithm [12]. The algorithm of [12], although also based on the OLS-LOO-LR regression framework, is very different from the current combined OLS-LOO-LR and MNQP algorithm. In particular, it transfers the kernels into the corresponding cumulative distribution functions and uses the empirical distribution function calculated on the training data set as the target function of the unknown cumulative distribution function. Moreover, in the work of [12], the unity constraint is met by normalising the kernel weight vector of the final selected model, which is nonoptimal, and the nonnegative constraint is ensured by adding a test to the OLS forward selection procedure, which imposes considerable computational cost.

The optimal kernel width was found to be $\rho = 0.65$ for the Parzen window estimate and $\rho = 1.2$ for both the previous SKD algorithm and the combined OLS-LOO-LR and MNQP algorithm, respectively, via cross validation. The results obtained by the three density estimator are summarised in Table 3. It can be seen that the proposed combined OLS-LOO-LR and MNQP algorithm yielded sparser kernel density estimates with better test performance.

5 Conclusions

A regression framework has been proposed for sparse kernel modelling, which unifies the supervised regression and classification problems as well as the unsupervised PDF learning problem. An OLS algorithm has been developed for selecting sparse kernel models that generalise well, based on the LOO test criteria and coupled with local regularisation. For sparse kernel density estimation, a combined approach of the OLS-LOO-LR algorithm and multiplicative nonnegative quadratic programming has been proposed, with the OLS-LOO-LR algorithm selecting a sparse kernel density estimate while the MNQP algorithm computing the kernel weights of the selected model to meet the constraints for density estimate. Empirical data modelling results involving regression, classification and density estimation have been presented to demonstrate the effectiveness of the proposed unified data modelling framework based on the OLS-LOO-LR algorithm, and the results shown have confirmed that this unified sparse kernel regression framework offers a state-of-the-art for data modelling applications.

References

1. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
2. Tipping, M.E.: Sparse Bayesian learning and the relevance vector machine. J. Machine Learning Research 1, 211–244 (2001)
3. Sha, F., Saul, L.K., Lee, D.D.: Multiplicative updates for nonnegative quadratic programming in support vector machines. Technical Report, MS-CIS-02-19, University of Pennsylvania, USA (2002)
4. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA (2002)
5. Vapnik, V., Mukherjee, S.: Support vector method for multivariate density estimation. In: Solla, S., Leen, T., Müller, K.R. (eds.) Advances in Neural Information Processing Systems, pp. 659–665. MIT Press, Cambridge (2000)
6. Girolami, M., He, C.: Probability density estimation from optimally condensed data samples. IEEE Trans. Pattern Analysis and Machine Intelligence 25(10), 1253–1264 (2003)
7. Chen, S., Billings, S.A., Luo, W.: Orthogonal least squares methods and their application to non-linear system identification. Int. J. Control 50(5), 1873–1896 (1989)
8. Chen, S., Cowan, C.F.N., Grant, P.M.: Orthogonal least squares learning algorithm for radial basis function networks. IEEE Trans. Neural Networks 2(2), 302–309 (1991)
9. Chen, S., Hong, X., Harris, C.J.: Sparse kernel regression modelling using combined locally regularized orthogonal least squares and D-optimality experimental design. IEEE Trans. Automatic Control 48(6), 1029–1036 (2003)
10. Chen, S., Hong, X., Harris, C.J., Sharkey, P.M.: Sparse modelling using orthogonal forward regression with PRESS statistic and regularization. IEEE Trans. Systems, Man and Cybernetics, Part B 34(2), 898–911 (2004)
11. Chen, S.: Local regularization assisted orthogonal least squares regression. Neurocomputing 69(4-6), 559–585 (2006)
12. Chen, S., Hong, X., Harris, C.J.: Sparse kernel density construction using orthogonal forward regression with leave-one-out test score and local regularization. IEEE Trans. Systems, Man and Cybernetics, Part B 34(4), 1708–1717 (2004)
13. Chen, S., Hong, X., Harris, C.J.: An orthogonal forward regression technique for sparse kernel density estimation. Neurocomputing (to appear, 2007)
14. Hong, X., Sharkey, P.M., Warwick, K.: Automatic nonlinear predictive model construction algorithm using forward regression and the PRESS statistic. IEE Proc. Control Theory and Applications 150(3), 245–254 (2003)
15. Hong, X., Chen, S., Harris, C.J.: Fast kernel classifier construction using orthogonal forward selection to minimise leave-one-out misclassification rate. In: Proc. 2nd Int. Conf. Intelligent Computing, Kunming, China, August 16-19, pp. 106–114 (2006)
16. Parzen, E.: On estimation of a probability density function and mode. The Annals of Mathematical Statistics 33, 1066–1076 (1962)
17. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapman Hall, London (1986)
18. Myers, R.H.: Classical and Modern Regression with Applications, 2nd edn. PWS Pub. Co., Boston, MA (1990)
19. http://www.ics.uci.edu/~mlearn/MLRepository.html
20. http://ida.first.fhg.de/projects/bench/benchmarks.htm
21. Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for AdaBoost. Machine Learning 42(3), 287–320 (2001)

Advanced Forecasting and Classification Technique for Condition Monitoring of Rotating Machinery

Ilya Mokhov[1] and Alexey Minin[2]

[1] Saint-Petersburg State University, Applied Mathematics and Control Processes
Ilya.Mokhov@siemens.com
[2] Saint-Petersburg State University, Applied Computational Physics
Alexey.Minin@siemens.com

Abstract. Prediction and classification of particular faults in rotating machinery, based on a given set of measurements, could significantly reduce the overall costs of maintenance and repair. Usually the vibration signal is sampled with a very high frequency due to its nature, thus it is quite difficult to do considerably long forecasting based on the methods, which are suitable for e.g. financial time series (where the sampling frequency is smaller). In this paper new forecasting and classification technique for particular vibration signal characteristics is proposed. Suggested approach allows creating a part of control system responsible for early fault detection, which could be used for preventive maintenance of industrial equipment. Presented approach can be extended to high frequency financial data for the prediction of "faults" on the market.

Keywords: fault analysis and prevention, artificial neural networks, artificial intelligence, rotating machinery, ball bearing failures, predictive monitoring.

1 Introduction

The faults in particular parts of industrial equipment could cause serious problems such as production losses, expensive repair procedures or even the personnel injures. Therefore the problem of fault analysis and prevention is very important. One of the main reasons of breakdowns of rotating machinery is the bearing failures. Plenty of papers, during the last decades, were devoted to the analysis of such kinds of faults by different methods of vibration analysis [11, 12, 13, 15]. The aim of this paper is to present the combination of forecasting and classification techniques, which could be used for the fault analysis and prevention.

In case the sampling rate is measured in kHz (for the dataset used in this paper sampling rate was equal to 40kHz), it is possible to estimate that prediction, achieved by the forecasting based on the pure time signal from the vibration sensors, will be only some microseconds ahead, which seems to be useless in terms of practical applications. Therefore in order to achieve applicable prediction for overcoming the problem described above should be introduced.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 37–46, 2007.

Let us consider the measurements obtained from rotating equipment containing ball bearings. Wide range of vibration analysis techniques could be used for analysis of the given measurements. The main idea of the traditional approaches is to analyze the peaks existence for particular frequencies and their multipliers (for more information regarding calculation of these frequencies and analysis methods [4, 5])

This paper will be devoted to the method which could be used for estimation of some particular defect frequency components evolution by using the artificial intelligence methods. The artificial neural networks, Neural Clouds [9] (further NC) and an advanced method of signal decomposition would be used in this research together with a trick of the particular fault related features extraction from Fourier spectrum.

It is out of the scope of this paper to describe the procedure of vibration data acquisition, for more information authors refer to [4]. In the following it would be assumed that a given time signal T was measured using vibration sensor for some ball bearing based system. The measurement strategy for efficient application of the presented method would be proposed in the following.

2 Preparation of Inputs for the Predictor

In this section data preprocessing and the preparation of the inputs for the forecasting and classification algorithms will be discussed. Following the work [6], it is a very important and challenging problem.

Let us divide given time sequence T into the set of equidistant sub sequences $\{ t_i \}$ so that the time interval between them is equal to $\widehat{t}$. For the generalization of presented approach this procedure corresponds to the process of collecting the number of measurements periodically. It should be mentioned that dealing with a limited amount of measurements one should always remember about the tradeoff between how far one wants to forecast (it would be shown that it is related to the time interval between consequent measurements) and how many patterns one needs to make the prediction.

For every subsequence t_i the Fourier spectrum F_i is computed. Having the set of "consequent" Fourier spectra $\{F_i\}_{i=\overline{1,N}}$ the frequency dynamics, if any, could be observed. So that the monitoring of the important changes in signal frequency characteristic over the time can be done. This approach is related to the time frequency analysis (see fig. 1). Each of the obtained spectra could be used for analysis of the bearing conditions for the corresponding time interval. The idea here is to do forecasting in order to get the estimation of future spectral characteristic. In order to do it the particular frequency component changes over the sequence of spectra would be considered.

The problem is that the spectrum itself contains too many data and, moreover, not all these data are used for particular faults detection. Therefore it would be consistent to extract few features of the great importance regarding the particular type of fault

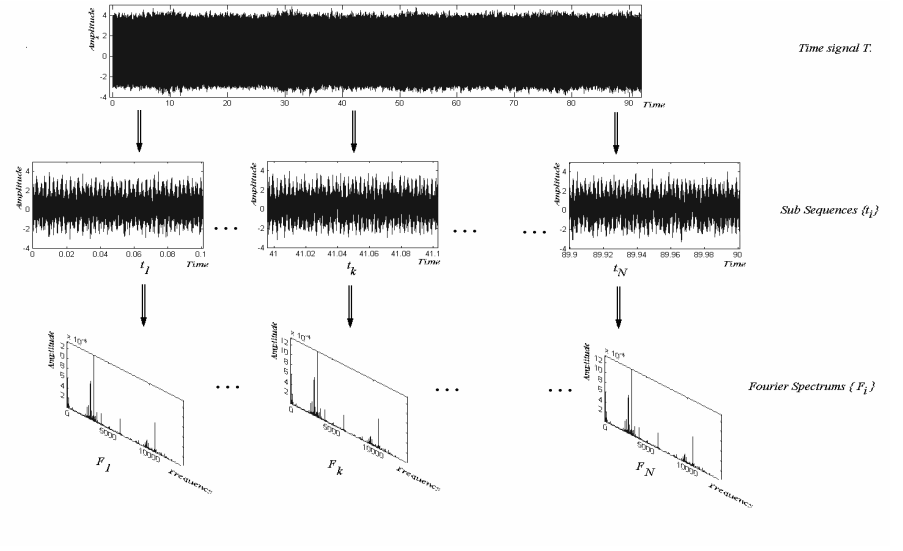


Fig. 1. Data preprocessing stage illustration

from the overall frequency data set and to do forecasting only for them (see fig.2.). As a possible feature here an analogue of a crest factor measure C_j^k (see eq.1) can be used.

$$C_j^k = \frac{peak\ value}{RMS} = \frac{\max\limits_i \{f_i\}}{\sqrt{\frac{\sum\limits_{i=1}^{W_j} f_i^2}{W_j}}},\ j = \overline{1,M}\ and\ k = \overline{1,N} \tag{1}$$

where f_i are the frequencies from selected window and M is the number of features for every spectrum and N is the number of spectra. For $\forall j \in [1, M]$ values $\left\{C_j^k\right\}_{k=\overline{1,N}}$ form time series with a sampling rate inversely proportional to the time interval $\widehat{t}$ introduced above.

The overall spectrum feature extraction scheme could be described as a selection of important frequencies (e.g. the defect frequencies or the frequencies of particular interest), choosing an appropriate window around each of the frequencies, and then calculating the mentioned above crest factor analogues values for those windows (see fig. 3). Situation when the selected measure is close to 1 corresponds to the absence of peak within the window considered, while relatively high value corresponds to the existence of peak respectively.

Finally as an input for the predictor a set of the time series $\left\{C_j^k\right\}_{k=\overline{1,N}}$ is used, each of them corresponds to the selected particular frequency component j and has a

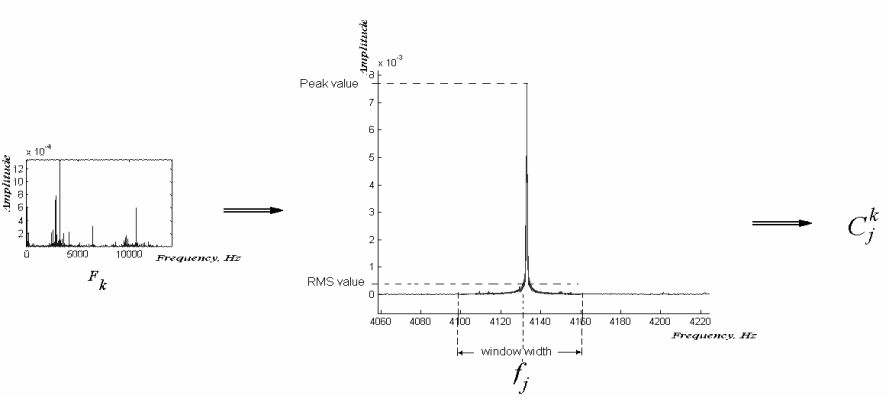


Fig. 2. Feature extraction idea

sampling rate, defined by the time interval $\hat{t}$ determined as a gap between two measurements. The aim of this process is to increase the time intervals between the data points, without loosing the important high frequency information.

One of the most important restrictions of the scheme presented is that all the measurements should be done within approximately similar conditions, such as rotation speed, external noise etc. Therefore one could expect that the defect growth will be observable within a selected frequency interval. Overcoming of this restriction is theoretically possible and could be considered as one of the further steps to extend the presented approach.

An example of the time series $\left\{C_j^k\right\}_{k=\overline{1,N}}$ is shown at the figure 3.

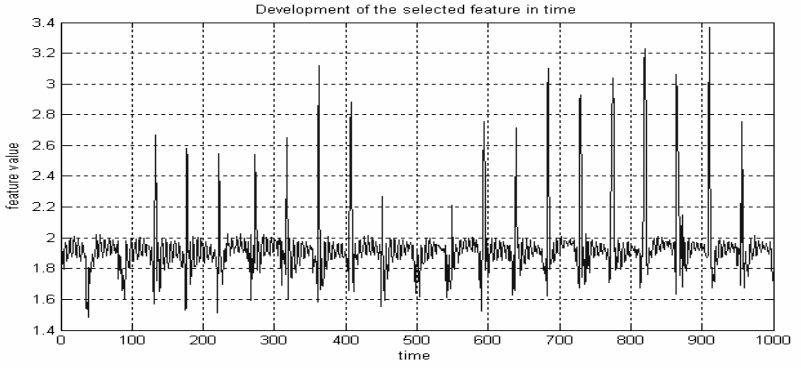


Fig. 3. Example of features time series we are going to predict

Then the Empirical Mode Decomposition [1, 2, 6] (further EMD) should be used in order to decompose initial signal into the set of orthogonal and non correlated time series (modes). The sum of the modes is equal to the initial signal (with an error of

order 10^{-16}). Let us assume that the next state of the system will depend mainly on the current state of the system and depend also on few previous states with smaller weights (see fig.4). Therefore, the intrinsic mode functions [1, 2] at the current moment t and the current state of the system with a few lagged intrinsic mode functions (with smaller weights) should be used as inputs for the ANN. The result is shown in the figure 4.

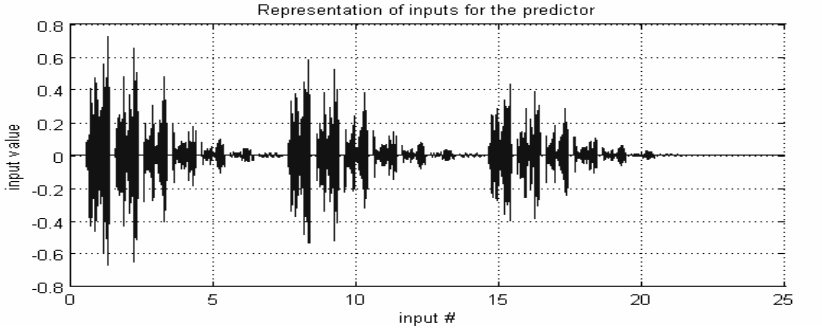


Fig. 4. Visualization of all inputs for ANN

As one can see from figure 4, some of the inputs (e.g. #6 and #7) are not of the great importance in comparison with the others. Nevertheless all of them have to be taken into account in order to avoid border effects of EMD method which could be rather strong [1]. Since the next state depends mainly on the current state (according to the assumption above) neural network have to be retrained to maintain the conditions of such specific process and to neglect border effects of EMD filter.

3 Forecasting Technique and Predictor Architecture

Due to the fact that constructed time series are very specific, the architecture of predictor should be chosen properly. There are two main properties which the predictor should have. First one is good approximation capabilities; second one is presence of embedded memory, since time series represent some dynamical model. So far instead of simple feed-forward artificial neural network (further ANN), namely perceptron, which stands for the static mapping from the input to the output space (see fig.5. left hand side) it would be more convenient to use recurrent ANN, namely Elman ANN [7] (see fig.5. right hand side), since it implies the dynamic memory. Elman networks are also well known for their ability to extract features from time series. Moreover, recurrent networks are more robust in sense of over fitting, since this type of ANN corresponds to the so called infinite impulse response filter and require less parameter to describe system dynamics. Feed forward network corresponds to finite impulse response filter and need more parameters for mapping and therefore can lead to over fitting. Further the comparison of three architectures, namely time delay neural networks, linear regression and recurrent networks will be presented.

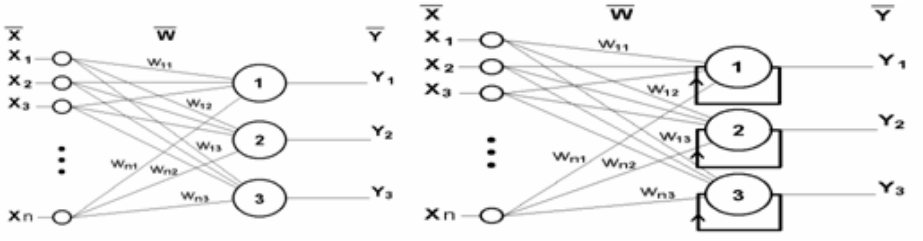


Fig. 5. $\{x_i\}$-input vector; $\{y_i\}$ – output vector; $\{w_{ij}\}$ – parameters (weights), Left hand side - Feed forward ANN; right hand side – RNN

All parameters of the network, such as a number of layers and number of neurons were obtained from computational experiment. Number of layers was chosen to be 2. In each layer the number of neurons was chosen to be 40. Then the neural network committee which consisted of 10 ANN's was constructed. The result over the committee was simple averaging of the each network outputs. The length of the data was chosen to be 1000 time steps (1000 points). Training algorithm for Elman network was chosen to be BFGS algorithm [10], for time delay neural network training algorithm was chosen to be Levenberg-Marquardt [14]. Results of the forecast are presented in figure 6.

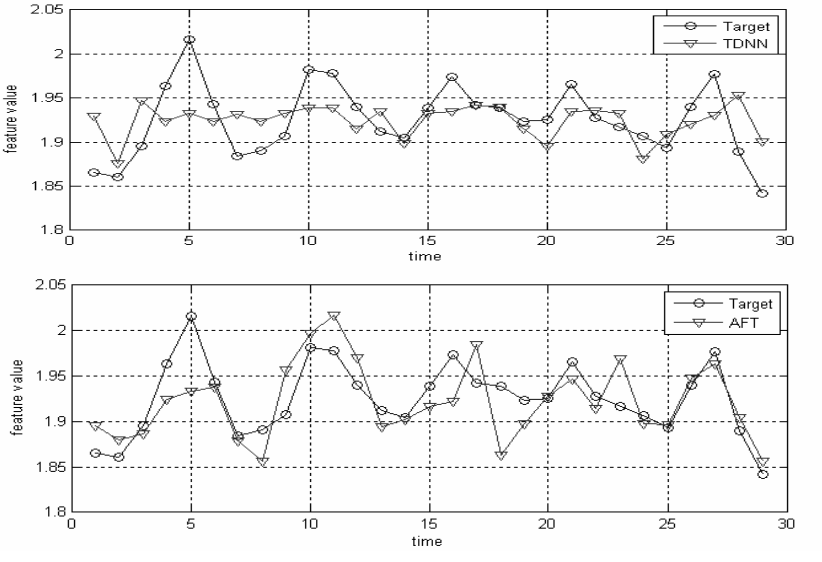


Fig. 6. One step forecast for the selected feature. The procedure was repeated 30 times to estimate statistical quality of prediction for one step.

As it could be noticed from figure 6 (bottom picture), non delayed forecast for the frequency component development (see table 1) has been obtained. Below the

comparison of linear regression and time delay neural network (further TDNN), with the proposed advanced forecasting technique based on RNN (further AFT) is given.

Table 1. Comparison of the quality of the forecast provided by different methods

	r	R^2	MSE
Linear regression	0,30	0,06	0,03
TDNN	0,38	0,14	0,02
AFT	0,66	0,30	0,01

Table 1 gives a brief overview of the quality of the forecast for the following statistics: mean squared error (MSE), determination coefficient (R^2, see eq.2) and correlation coefficient (r).

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y}_i)}, \tag{2}$$

where $\hat{y}_i$ - is a forecast, y_i - is a true value and $\overline{y}_i$ - is an average over all training patterns. As one can see from table 1 AFT has a better correlation with the target than TDNN, and R^2 measure shows that determination of the forecast provided by AFT is better. The bigger determination means the lower probability of the forecast delay. Moreover using R^2 measure the optimal number of neurons in hidden layer could be estimated. The optimal number of neurons corresponds to the saturation of R^2 measure, so that the increasing of the hidden layer does not improve the forecast.

The forecast was performed in a same manner for all features being extracted from Fourier spectrum, so that for all selected signal characteristics the future values were obtained. By analyzing the obtained values future machine state could be estimated. To make the procedure of classifying the machine condition as normal or abnormal automatic, the usage of a novel classification technique would be proposed in next section.

4 Classification Technique and Overall Scheme

To make process fully automatic and suitable for implementation of fault prevention system it was decided to add mechanism for classification. For this purpose the novel technique which is called “Neural clouds” [9] was chosen. Since the details of NC are not published at the moment, its main ideas will be briefly discussed in the following.

The concept which stands behind the NC term consists of creating an efficient data encapsulation mechanism for the so called one-side classifier, using the advanced clustering algorithm and extended Radial Basis functions network approach. The basic idea of the one-side classification usage in field of condition monitoring, fault analysis and prevention is that the real industrial data, which could be collected on the

plant, usually correspond to the normal conditions, while the bad data collection is expensive, and fault modeling is not always available. The system is trained on a given dataset which corresponds to the normal operating conditions of the plant and stands for the detection of the fluctuations from these conditions.

The simple example of classification is shown in figure 7. Here the data points are displayed with circles and confidence levels are displayed with lines. The plateau on the fig.7 right hand side corresponds to the conditions which suppose to be normal with probability equal to 1 and the slopes shows how the probability decreases down to 0.

After obtaining a new forecast one just "drops" or project it on the classifier (see fig.7.) and chooses the nearest confidence level. Then one should decide whether

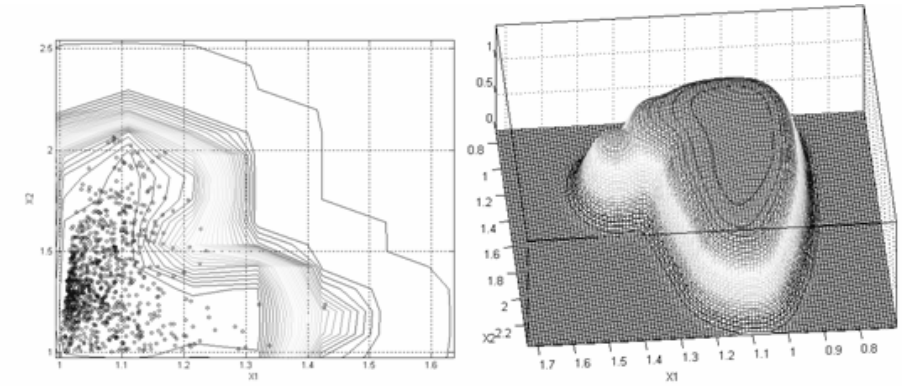


Fig. 7. Left hand side: Classification in 2D space for 2 selected features (normalized density), right hand side: classifier in 3D space (normalized density)

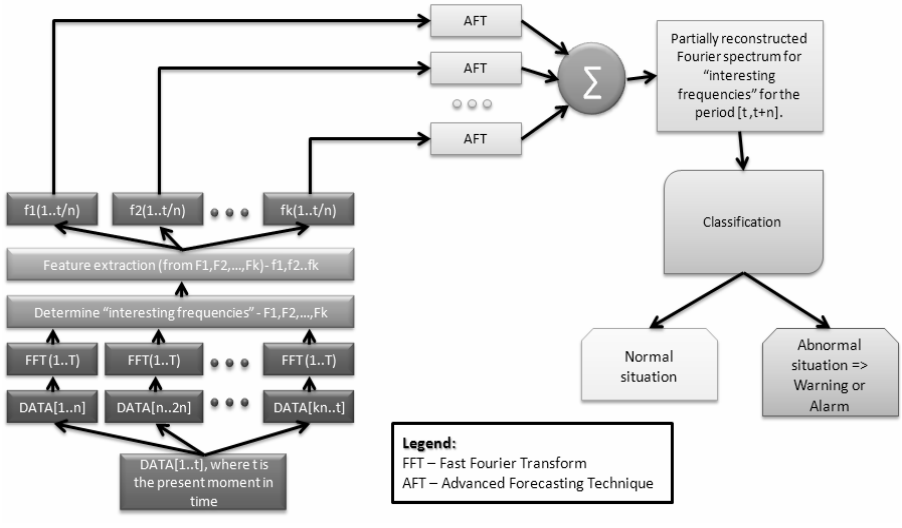


Fig. 8. Overall scheme of the system

forecasted state induces alarm or not, using the simple formula: $P_f = |1 - Conf|$, where P_f is probability of failure and $Conf$ is confidence level.

The overall scheme of the system is like following: first extract features from Fourier spectrum, then comes AFT to provide forecast of particular Fourier spectrum features and NC to recognize whether the predicted state is normal or abnormal. After the system is trained in back testing mode, AFT has to be retrained after each step, while NC remains the same if the normal conditions do not change. Overall scheme is shown in figure 8.

5 Conclusions and Outlook

The most of the neural network applications in the field of vibration diagnostic are devoted to the classification problem. In the presented paper authors made an attempt to apply it to the forecasting problem. The approach presented above could be considered as an additional part of the vibration analysis and monitoring system. System offers the experts a possibility to analyze the evolution of the particular defect, introduced by the bearings components and to perform predictive monitoring of the future state of the system. Additional classification stage makes the abnormal behavior detection process automatic. Together with a suitable measurement strategy it could be used for a considerably long term prediction of frequency characteristics of the vibration signal and therefore the prediction of the system conditions.

According to the set of experiments it was found that the proposed technique could be used for the prediction of the introduced frequency features.

As a possible extension of the suggested approach a forecasting for the additional values could be considered (e.g. forecasting enveloping spectrum features, some additional measurements and calculated statistical values (for instance kurtosis, skewness)). This will allow covering the forecasting of the main characteristics of vibration signal used for the analysis of the equipment conditions. Taking into account changing rotation speed one should consider as the thing of a great importance simplifying the data acquisition process.

The extension of the forecasting horizon to middle range could be considered on the basis of the paper [8].

Acknowledgments. The work was done with support of OOO “Siemens” CT, namely Bernhard Lang (Bernhard.lang@siemens.com), “Fault analysis and prevention” and with support of Kuperin Yuri (Yuri.Kuperin@gmail.com), Saint-Petersburg State University.

References

1. Rilling, G., Flandrin, P., Gonëcalv‘es, P.: On empirical mode decomposition and its algorithms. In: IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing NSIP-03, Grado (I)

2. Huang, N.E., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N.-C., Tung, C.C., Liu, H.H.: The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Mathematical, Physical and Engineering Sciences 454(1971), 903–995 (1998)
3. Magrin-Chagnolleau, I., Baraniuk, R.G.: Empirical mode decomposition based on it's time-frequency attributes. In: Proceedings of the 69th SEG Meeting, Rice University, Houston, Texas
4. Barkova, N.A.: Introduction to Vibration Diagnostics of Rotating Machinery and Equipment. In: SPbSMTU (in Russian) (2003)
5. Barkov, A.V., Barkova, N.A., Azovcev, A.U.: Monitoring and Diagnostics of Rotor Machines by Vibration. In: SPbSMTU (in Russian) (2000)
6. Minin, A., Kuperin, Y., Lang, B.: Increasing the quality of neural forecasts with the help of Empirical Mode Decomposition. In: Proceedings of the International Scientific Conferences "Intelligent systems" (AIS 2007) and "Intelligent CAD's" (CAD-2007), Physmathlit, vol. 4, pp. 68–74 (2007)
7. Narendra, K.S., Parthasarathy, K.: Identification and control of dynamical systems using neural networks. IEEE Trans. Neural Networks 1, 4–27 (1990)
8. Ludd, K.: Michael Small Towards long-term predictions. Physica D 136, 31–44 (2000)
9. Lang, B., Mokhov, I.: Neural Clouds for monitoring of complex plant conditions (to appear)
10. Broyden, C.G.: Journal of the Institute of Mathematics and Its Applications 6, 76–90 (1970)
11. Sanza, J., Pererab, R., Huertab, C.: Fault diagnosis of rotating machinery based on auto-associative neural networks and wavelet transforms. Journal of Sound and Vibration 302, 981–999 (2007)
12. Hoffman, A.J., van der Merwe, N.T.: The application of neural networks to vibration diagnostics for multiple fault conditions. Computer Standards & Interfaces 24, 139–149 (2002)
13. Kalkat, M., Yildirim, S., Uzmay, I.: Design of artificial neural networks for rotor dynamics analysis of rotating machine systems. Mechatronics 15, 573–588 (2005)
14. Hagan, M.T., Menhaj, M.: Training feed forward networks with the Marquardt algorithm. IEEE Transactions on neural networks 5(6), 989–993
15. Howard, I.: A review of rolling element bearing vibration "Detection, diagnosis and prognosis", DSTO Aeronautical and Maritime Research Laboratory (1994)

Out of Bootstrap Estimation of Generalization Error Curves in Bagging Ensembles

Daniel Hernández-Lobato, Gonzalo Martínez-Muñoz, and Alberto Suárez

Escuela Politécnica Superior,
Universidad Autónoma de Madrid,
C/ Francisco Tomás y Valiente, 11, Madrid 28049 Spain
{daniel.hernandez, gonzalo.martinez, alberto.suarez}@uam.es

Abstract. The dependence of the classification error on the size of a bagging ensemble can be modeled within the framework of Monte Carlo theory for ensemble learning. These error curves are parametrized in terms of the probability that a given instance is misclassified by one of the predictors in the ensemble. Out of bootstrap estimates of these probabilities can be used to model generalization error curves using only information from the training data. Since these estimates are obtained using a finite number of hypotheses, they exhibit fluctuations. This implies that the modeled curves are biased and tend to overestimate the true generalization error. This bias becomes negligible as the number of hypotheses used in the estimator becomes sufficiently large. Experiments are carried out to analyze the consistency of the proposed estimator.

1 Introduction

In many classification tasks, bagging [1] improves the generalization performance of individual base learners. However, due to need of repeated executions of the underlying algorithm, the computational requirements to estimate generalization error of this algorithm by traditional statistical techniques, such as cross validation, can be quite expensive. In order to address this problem we investigate the properties of an efficient estimator based on the Monte Carlo approach to ensemble learning developed in [2,3,4]. Assuming that the probability of selecting a hypotheses that misclassifies a given instance is known, the average error on that instance of a Monte Carlo ensemble of arbitrary size can be computed in terms of the binomial distribution [2,3,4]. Using this analysis, it is possible to model error curves that describe the error of the ensemble as a function of the number of predictors in the ensemble. In this work we propose an out of bootstrap estimator for the generalization error of a bagging ensemble based on computing the misclassification probabilities on out of bootstrap data. The estimator is shown to be biased. Nonetheless, the bias component decreases as the size of the ensemble used to perform estimations grows.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 47–56, 2007.

2 Monte Carlo Ensemble Learning

Monte Carlo (MC) algorithms [5,2,3,4] provide a useful framework for the analysis of learning ensembles. In order to introduce some notation and basic concepts, we provide a brief review of Monte Carlo algorithms applied to classification problems.

A Monte Carlo algorithm is a stochastic system that returns an answer to an instance of a problem with a certain probability. The algorithm is consistent if does not generate two different correct answers to the same problem instance. Different executions of the algorithm are assumed to be statistically independent, conditioned to some known information (in classification, this known information is the training data). A Monte Carlo algorithm is said to be α-*correct* if the probability that it gives a wrong answer to a problem instance is at most $p = 1-\alpha$. The *advantage* of such an algorithm is defined to be $\gamma = \alpha - \frac{1}{2} = \frac{1}{2} - p$. The accuracy of a consistent Monte Carlo (MC) algorithm with positive *advantage* can be amplified to an arbitrary extent simply by taking the majority answer of repeated independent executions of the algorithm. In B independent executions of the algorithm, the probability of b failures follows a binomial distribution

$$Pr(b) = \binom{B}{b} p^b (1-p)^{B-b}. \tag{1}$$

Assuming that B is odd, the answer of the amplification process would be wrong only if more than half of the responses of the base algorithm were wrong. The probability of such an event is

$$\pi(p, B) = \sum_{b=\lfloor \frac{B}{2} \rfloor + 1}^{B} \binom{B}{b} p^b (1-p)^{B-b}. \tag{2}$$

If $p < \frac{1}{2}$ and $B \to \infty$ (2) tends to 0 or, equivalently, the probability of a correct output from the algorithm tends to one. On the other hand, if $p > \frac{1}{2}$, the algorithm does not asymptotically produce a correct answer.

Consider a binary classification learning problem characterized by the fixed joint probability distribution $\mathcal{P}(\mathbf{x}, y)$, where $\mathbf{x} \in \mathcal{X}$,and $y \in \mathcal{Y} = \{-1, +1\}$. For simplicity, $\mathcal{X}$ is assumed to be discrete and finite with cardinality N. This in turn implies that the space of hypothesis $\mathcal{H}$ is also finite with cardinality J. The results can be readily extended to continuous infinite spaces. In these conditions Table 1 summarizes the performance of a set of hypotheses $\mathcal{H}$. The nth row in this table corresponds to the nth vector $\mathbf{x}_n \in \mathcal{X}$, which has a probability $\mathcal{P}(\mathbf{x}_n)$. The jth column corresponds to the jth hypothesis $h_j \in \mathcal{H}$, which has a probability q_j of being applied. The element $\xi_j(\mathbf{x}_i) \in \{0, 1\}$ at row i and column j of the inner matrix is an indicator whose value is 1 if hypothesis h_j misclassifies instance $\mathbf{x}_i$ and 0 otherwise.

To classify instance $\mathbf{x}$, the Monte Carlo algorithm defined in [4] proceeds by selecting one hypothesis h_j from $\mathcal{H}$ with probability q_j. It then assigns the class

label $h_j(\mathbf{x}) \in \mathcal{Y}$. Elements on the right-most column of Table 1 are defined as

$$p(\mathbf{x}_i) = \sum_{j=1}^{J} q_j \xi_j(\mathbf{x}_i), \tag{3}$$

where $p(\mathbf{x}_i)$ is the probability of extracting a hypothesis that misclassifies instance $\mathbf{x}_i$. With this definition, the algorithm is $(1 - p(\mathbf{x}_i))$-*correct* on $\mathbf{x}_i$. If $p(\mathbf{x}_i) < \frac{1}{2}$ then, the *advantage* of the algorithm on instance $\mathbf{x}_i$ is strictly positive. This means that we can amplify the answer to this instance by running the algorithm B times and taking a majority vote among the classifications generated. However, if $p(\mathbf{x}_i) > \frac{1}{2}$ this same procedure would actually worsen the results and make the probability of generating a right answer tend to zero.

If all the hypotheses in $\mathcal{H}$ are available, the classification produced after B executions of the algorithm is a random variable whose average is

$$H(\mathbf{x}) = sign\left(\sum_{j=1}^{J} q_j h_j(\mathbf{x})\right). \tag{4}$$

As $B \to \infty$ the distribution of this random variable becomes more peaked around this mean. The expected error of the Monte Carlo algorithm is

$$\mathcal{E}(B) = \sum_{\mathbf{x} \in \mathcal{X}} \pi\left(p(\mathbf{x}), B\right) \mathcal{P}(\mathbf{x}), \tag{5}$$

where $\pi\left(p(\mathbf{x}), B\right)$ is given by (2). The limit of the error as B approaches ∞ is

$$\mathcal{E}_\infty = \lim_{B \to \infty} \mathcal{E}(B) = \sum_{\mathbf{x} \in \mathcal{X}_A} \mathcal{P}(\mathbf{x}), \tag{6}$$

where $\mathcal{X}_A = \{\mathbf{x} \in \mathcal{X} : p(\mathbf{x}) > \frac{1}{2}\}$ is the set of instances over which the algorithm cannot be amplified.

As noted in [4] bagging and the Monte Carlo algorithm we have just described are closely related. Assume that a labeled training dataset $\mathcal{T}^{(tr)} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, N_{tr}\}$ is available. Suppose that $\mathcal{H}$ is the set of hypotheses that can be generated by training a base learner on independent bootstrap samples extracted

Table 1. Elements in a Monte Carlo Ensemble Algorithm

	h_1	h_2	$\cdots$	h_J	
	q_1	q_2	$\cdots$	q_J	
$\mathbf{x}_1$	$\xi_1(\mathbf{x}_1)$	$\xi_2(\mathbf{x}_1)$	$\cdots$	$\xi_J(\mathbf{x}_1)$	$p(\mathbf{x}_1)$
$\mathbf{x}_2$	$\xi_1(\mathbf{x}_2)$	$\xi_2(\mathbf{x}_2)$	$\cdots$	$\xi_J(\mathbf{x}_2)$	$p(\mathbf{x}_2)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$\mathbf{x}_N$	$\xi_1(\mathbf{x}_N)$	$\xi_2(\mathbf{x}_N)$	$\cdots$	$\xi_J(\mathbf{x}_N)$	$p(\mathbf{x}_N)$

from the original training data. Bagging can be described as a Monte Carlo algorithm that first draws B hypotheses from $\mathcal{H}$ at random using a uniform probability distribution, and then uses the same B hypotheses to classify all data instances. From a statistical point of view, when classifying a single instance $\mathbf{x}$, the Monte Carlo algorithm described and bagging are equivalent. This observation means that Table 1 can also be used to analyze the generalization properties of bagging.

In particular, it can be shown [4] that the expected error of bagging with B hypotheses is given by (5). This expression provides a model for the error curves of bagging ensembles. These curves display the dependence of the classification error as a function of the ensemble size. In [4] the test and train error curves are modeled using (5), where the values of $p(\mathbf{x})$ and $\mathcal{P}(\mathbf{x})$ are estimated on the training and test samples, respectively. In the present investigation it is shown that the generalization error curves can be can be modeled using information only from the training data by computing bootstrap estimates of $p(\mathbf{x})$ in (5).

3 Error Curves for Bagging Ensembles

Ensemble methods such as bagging [1] have demonstrated their potential for improving the generalization performance of induced classifier systems. The success of bagging is related to its ability to increase the accuracy of a (possibly weak) learning algorithm $\mathcal{A}$. Bagging constructs a set of different hypotheses $\mathcal{H} = \{h_m; m = 1, 2, \ldots, M\}$ using in the learning algorithm $\mathcal{A}$ different surrogate training sets $\left\{\mathcal{T}_m^{(tr)}; m = 1, 2, \ldots, M\right\}$ obtained by bootstrap sampling from the original training data $\mathcal{T}^{(tr)}$ [6]. Provided that the base learning algorithm is unstable with respect to modifications in the training data, this procedure has the effect of generating a set of diverse hypotheses. Each instance is then classified by using majority voting scheme. If the errors of the different base learners are not fully correlated, the composite hypothesis should have a lower error than the individual hypotheses. Experimental analysis of bagging is given in [7,8,9,10].

As described in Section 2, the dependence of the classification error of a bagging ensemble on its size can be modeled within the framework of Monte Carlo theory for ensemble learning. The analysis is based on averages computed using the elements of Table 1. Assume that $\mathcal{H}$ is the set of hypotheses included in a bagging ensemble of size M. The classification error of a bagging ensemble of size B on a given dataset $\mathcal{T} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ can be estimated using

$$\mathcal{E}_{\mathcal{T}}(B) = \frac{1}{|\mathcal{T}|} \sum_{\mathbf{x}_i \in \mathcal{T}} \pi(\hat{p}(\mathbf{x}_i), B), \tag{7}$$

which corresponds to (5) with $\mathcal{P}(\mathbf{x})$ replaced by the empirical distribution of the examples in $\mathcal{T}$ and with the value of $p(\mathbf{x}_i)$ estimated on $\mathcal{H}$ as

$$\hat{p}(\mathbf{x}_i) = \frac{1}{M} \sum_{m=1}^{M} \xi_m(\mathbf{x}_i). \tag{8}$$

The indicator $\xi_m(\mathbf{x}_i)$ is the error of each hypothesis $h_m \in \mathcal{H}$ on instance $\mathbf{x}_i \in \mathcal{T}$. Note that B and M can be different. That is, one can use the hypothesis in the bagging ensemble of size M to estimate the generalization error of an ensemble of arbitrary size B. Because (7) has a smooth dependence on B it can be used to estimte the convergence level of bagging with B hypotheses. In fact one can take $B \to \infty$ to approximate the asymptotic limit of the error of bagging.

3.1 Bias Analysis

The estimator of the ensemble error $\mathcal{E}_{\mathcal{T}}(B)$ given by (7) is biased because the value of M, the number of hypotheses used to estimate $p(\mathbf{x}_i)$, is finite. In contrast with B, whose value can be made arbitrarily large, M is at most as large as the size of the bagging ensemble constructed. The dependence of the bias of $\mathcal{E}(B)$ with M can be estimated within the Monte Carlo framework. The value computed in (8) using a set of hypothesis of finite size M is a realization of a random variable $\hat{p}(\mathbf{x}_i)$ that follows a binomial distribution with parameter $p(\mathbf{x}_i)$. The average of (7) over this random variable is

$$\mathbb{E}_{\hat{\mathbf{p}}(\mathbf{x})}\left[\mathcal{E}_{\mathcal{T}}(B)\right] = \frac{1}{|\mathcal{T}|}\sum_{i=1}^{|\mathcal{T}|}\mathbb{E}_{\hat{p}(\mathbf{x}_i)}\left[\pi(\hat{p}(\mathbf{x}_i), B)\right], \tag{9}$$

where, for an ensemble of size M,

$$\mathbb{E}_{\hat{p}(\mathbf{x}_i)}\left[\pi(\hat{p}(\mathbf{x}_i), B)\right] = \sum_{m=0}^{M}\binom{M}{m}p(\mathbf{x}_i)^m(1-p(\mathbf{x}_i))^{M-m}\pi(\frac{m}{M}, B), \tag{10}$$

As a result of the non linearity of (2), the value of (10) need not be equal to $\pi(p(\mathbf{x}_i), B)$. Fig. 1 (left) illustrates this effect. The discontinuous curves correspond to (10) and display the expected value of the estimator of the ensemble error on a single instance as a function of $p(\mathbf{x}_i)$ for different values of M. The continuous line plots the $M \to \infty$ limit of the discontinuous curves, which corresponds to $\pi(p(\mathbf{x}_i), B)$. The graphs are drawn for an ensemble of size $B = 1001$. Similar results are obtained for different values of B. In the limit $B \to \infty$ the $M = 1$ curve remains unchanged (a straight line) and the $M \to \infty$ curve tends to a step function.

This figure illustrates that for $M > 1$ and a fixed value of B the Monte Carlo amplification is more effective the further the value of $p(\mathbf{x}_i)$ is away from $\frac{1}{2}$. For a given value of $p(\mathbf{x}_i)$, the bias of the finite M estimate is the vertical distance between the corresponding (discontinuous) line and the continuous one. The smaller the values of M the larger the variance of $\hat{p}(\mathbf{x}_i)$, and, in consequence, the larger the bias of the estimator. The sign of the bias is positive for $p(\mathbf{x}_i) < \frac{1}{2}$ and negative for $p(\mathbf{x}_i) > \frac{1}{2}$. Since examples correctly classified by the ensemble of size M have $p(\mathbf{x}_i) < \frac{1}{2}$ and incorrectly classified examples have $p(\mathbf{x}_i) > \frac{1}{2}$, some bias cancellation should be expected when computing (7). Assuming that the ensemble error rate is smaller than 1/2, the total bias for finite M is typically positive.

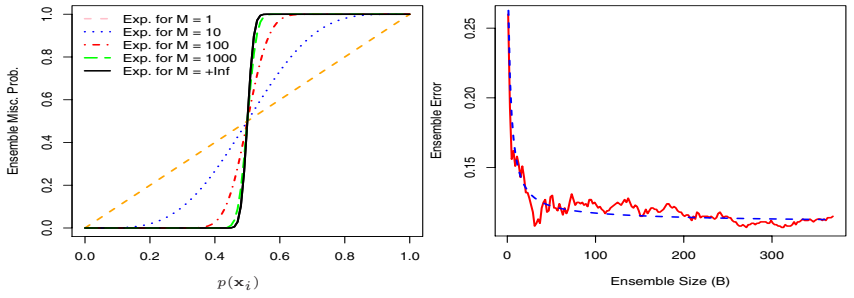


Fig. 1. (left) Expected value of $\pi(\rho(\mathbf{x}_i), B)$ as a function of the true misclassification probability $p(\mathbf{x}_i)$ for $B = 1001$ and different values of M. Discontinuous lines correspond to finite values of M. The continuous line corresponds to the $M \to \infty$ curve, $\pi(p(\mathbf{x}_i), B)$. (right) Ensemble error measured over a test set (continuous) and out of bootstrap estimation computed by means of (7) and (11) (discontinuous).

As a result of the reduction in the variance of $\hat{p}(\mathbf{x}_i)$ the bias component can be made arbitrarily small provided that sufficiently large ensembles are used. Therefore the estimator (7) is consistent in the limit $M \to \infty$. In particular, if one wishes to estimate the error of a subensemble composed of $B \leq M$ different hypotheses extracted at random from the original ensemble of size M one should use all the M elements in the original ensemble to compute (8).

3.2 Out of Bootstrap Estimation

In this section we propose an out of bootstrap estimator for the generalization error of bagging ensembles of arbitrary sizes. In bootstrap sampling examples are selected at random from the original set with replacement. On average, 36.8% of the extractions in a bootstrap sample of the same size as the original set correspond to repeated elements. This means that there are 36.8% examples in the original set which are not present in a particular bootstrap sample. Out of bootstrap techniques take advantage of these data to perform estimations of the generalization properties of the predictors constructed with the bootstrap sample.

Estimates of the generalization error of bagging ensembles based on out of bootstrap data have been considered in [11,12]. In [11] a *bias-variance* decomposition of the generalization error of bagging ensembles for regression problems is carried out. Out of bootstrap data is used to estimate the bias component of the error, which is equal to the asymptotic error of the ensemble. In [12] out of bootstrap data is used to estimate the generalization error of bagging ensembles. The classification error for a given instance in the original training set is estimated using only the classifiers trained with bootstrap samples that do not include such instance (on average, 36.8% of the total ensemble members). The generalization error of the ensemble is obtained by averaging these error estimates for single instance over the whole training set. Notice that this procedure provides only a single estimate, while the estimator proposed in the current article models the

complete error curve. The estimator proposed in [12] is a particular case of the one given in the present work. Breiman's estimator is recovered when all the hypotheses in the ensemble are used for the estimation of (8), and the asymptotic limit $B \to \infty$ of (7) is taken.

Error curves estimated on the training set typically underestimate the true generalization error. To avoid this training bias, it is possible to give an estimate $\mathcal{E}_{VAL}(B)$, where (8) is computed using a validation set independent of the training data. Alternatively, an out of bootstrap estimator $\mathcal{E}_{OB}(B)$ that uses only training data can be designed. For each instance in the training set $\mathbf{x}_i \in \mathcal{T}^{(tr)}$, $p(\mathbf{x}_i)$ is estimated as the average of $\xi_m(\mathbf{x}_i)$ over the set of hypotheses trained on bootstrap samples that do not include $\mathbf{x}_i$

$$\hat{p}(\mathbf{x}_i) = \frac{1}{\left|\mathcal{H}_{\setminus i}\right|} \sum_{h_m \in \mathcal{H}_{\setminus i}} \xi_m(\mathbf{x}_i), \tag{11}$$

where $\mathcal{H}_{\setminus i} = \{h_m : h_m \in \mathcal{H}, (\mathbf{x}_i, y_i) \notin \mathcal{T}_m^{(tr)}\}$. The set $\mathcal{T}_m^{(tr)}$ is the bootstrap sample of $\mathcal{T}^{(tr)}$ used to train h_m. On average $\mathcal{H}_{\setminus i}$ contains 36.8% of the initial hypotheses in bagging . The out of bootstrap estimate proposed in [12] corresponds to the limit $B \to \infty$ and is given by (6) with $p(\mathbf{x}_i)$ estimated by (11).

Fig. 1 (right) displays generalization error curves of a bagging classification ensemble for the synthetic problem *Twonorm*, as a function of its size. The ensemble is trained using $N_{tr} = 300$ labeled instances. The continuous line traces the actual error on an independent test set with $N_{test} = 1000$ elements. The dashed line corresponds to $\mathcal{E}_{OB}(B)$ estimated on $M = 370$ bagging hypotheses using out of bootstrap data. Note that the proposed out of bootstrap estimator (7) has a smooth dependence on B.

4 Experiments

In order to assess the reliability of the proposed out of bootstrap estimator experiments are carried out in several real world and synthetic binary classification problems from the UCI repository [13] (see Table 2). Each real world problem data set is split into three subsets: train, validation and test. The size of the training set is set to $\frac{4}{9}$ of the total data while the size of the validation and test set are set to $\frac{2}{9}$ and $\frac{1}{3}$ respectively. For the synthetic problems *Twonorm* and *Ringnorm*, train, test and validation sets are randomly built as described in Table 2. The validation set is used to provide an independent check on whether using out of bootstrap data has an undesired effect in the estimation of the misclassification probabilities $\hat{p}(\mathbf{x}_i)$.

The experimental protocol consist of the following steps:

(i) Data examples are partitioned at random into train, validation and test sets.
(ii) A bagging ensemble of 1000 CART trees [14] is built using the training set.
(iii) Estimates of the error by the procedure described in Section 3.2 are computed for subensembles of different sizes ($B = 1, 2, \ldots, 1000$). A first set of

Table 2. Description of the problems and data sets used in the experiments

Problem	Train	Test	Val.	Classes
Ringnorm	300	1000	300	2
Twonorm	300	1000	300	2
Sonar	63	69	49	2
Ionosphere	156	117	78	2
Breast	310	233	155	2
Pima	341	256	171	2

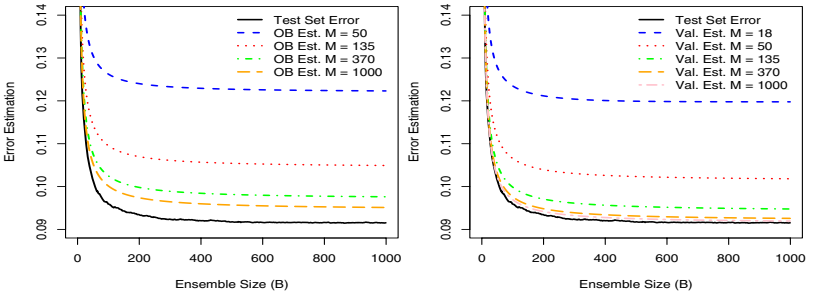


Fig. 2. Average ensemble error as a function of the ensemble size for the classification problem *Twonorm*. Plotted curves depict test set error alongside with out of bootstrap (left-hand side) and validation (right-hand side) estimates of the generalization error for different values of M.

Table 3. Averages and standard deviations of the validation and out of bootstrap estimates of the generalization error and test errors (in %) for bagging ensembles of size $B = 1000$.

		Ringnorm	Twonorm	Sonar	Ionosphere	Breast	Pima
$M = 18$	VAL	14.8±3.4	12.0±3.2	**28.5±7.2**	10.6±3.7	**5.3±2.1**	**25.6±3.2**
$M = 50$	OB	14.9±1.9	12.2±2.1	28.8±5.1	10.5±2.2	**5.2±1.0**	**25.5±2.1**
	VAL	13.2±3.4	10.2±3.2	**27.9±7.1**	**10.3±3.8**	**5.1±2.0**	**25.5±3.1**
$M = 135$	OB	13.5±2.0	10.5±2.3	**27.9±4.8**	**10.2±2.1**	**5.1±0.9**	**25.4±2.0**
	VAL	**12.8±3.2**	**9.5±3.1**	**27.6±6.9**	**10.21±3.8**	**5.1±1.9**	**25.4±3.2**
$M = 368$	OB	12.9±2.0	9.8±2.3	**27.8±4.8**	**10.1±2.2**	**5.0±0.9**	**25.3±2.0**
	VAL	**12.6±3.3**	**9.3±3.2**	**27.5±6.8**	**10.1±3.8**	**5.1±1.9**	**25.4±3.1**
$M = 10^3$	OB	**12.8±2.0**	**9.5±2.6**	**27.7±4.7**	**10.1±2.2**	**5.0±0.9**	**25.3±2.0**
	VAL	**12.5±3.3**	**9.2±3.1**	**27.4±6.8**	**10.1±3.7**	**5.1±1.9**	**25.4±3.2**
Test Error		12.4±3.1	9.2±3.0	27.7±5.7	9.9±3.2	5.1±1.6	25.4±2.5

out of bootstrap estimators (OB) that use out of bootstrap data is built using a random selection of $M = 50$, $M = 135$, $M = 368$ and $M = 1000$ trees from the ensemble generated in (ii). A second set of validation

estimators (VAL) is constructed using validation data and a random selection of $M = 18$, $M = 50$, $M = 135$, $M = 368$ and $M = 1000$ trees from the ensemble built in (ii). Note that the out of bootstrap estimate effectively uses only 36.8% of the classifiers to estimate a given value of $p(\mathbf{x}_i)$. This means that the out of bootstrap estimator with M_{OB} trees should be compared with the validation estimator that uses $M_{VAL} \approx 0.368\, M_{OB}$ trees, so that both estimators are computed on the same *effective number* of hypotheses. In fact, the values $M_1 = 18$ $M_2 = 50$, $M_3 = 135$, $M_4 = 368$ and $M_5 = 1000$ are chosen so that $M_{i-1} = round(0.368\, M_i)$, starting from $M_5 = 1000$.

(iv) Finally, the error in the test set is calculated for subensembles containing the first B elements of the bagging ensemble generated in (ii), with $B = 1, 2, \ldots, 1000$.

The curves plotted and figures reported correspond to averages over 500 iterations of the steps (i)-(iv) for each problem.

Fig. 2 depicts the ensemble error as a function of ensemble size ($B = 1, 2, \ldots, 1000$) for the classification problem *Twonorm*. The continuous lines correspond to test set errors. The discontinuous lines are out of bootstrap (on the left-hand side) and validation estimates (on the right-hand side) of the generalization error with different values of M. Note that, in agreement with the results of Section 3.1, the bias of the Monte Carlo estimators in (2) becomes smaller as M increases and is fairly small for $M = 1000$ in all problems. As predicted, the error curves for the pairs $M_{OB} = round(0.368\, M_{VAL})$ are very similar. Finally, we point out that the bias of the estimator is typically positive. This is because, on average, the misclassification probabilities of the base learners over the problem instances are smaller than $\frac{1}{2}$ as shown in Section 3.1. The error curves for the other classification problems exhibit similar features.

Table 3 summarizes the values for the different estimators of the ensemble error with $B = 1000$ and different values of M. The values tabulated are the mean and standard deviation over 500 executions carried out with different random partitions of the data. The average and standard deviation of the error on the test set are displayed in the last row of the table. These results illustrate that for sufficiently high values of M the out of bootstrap method provides a consistent estimate for the generalization error of the ensemble. The values displayed in boldface correspond to cases in which the difference between the expected value of the error estimate and the actual test error is not statistically significant at a confidence level of 1%.

As expected, for $M_{VAL} = M_{OB}$ the validation estimator is more accurate than the out of bootstrap one. This behavior is particularly noticeable in the synthetic problems *Ringnorm* and *Twonorm*. However, the average estimates for $M_{VAL} = 0.368\, M_{OB}$ are similar. Variances are roughly independent of M. They tend to be smaller for $\mathcal{E}_{OB}$ because of the presence of correlations between the out of bootstrap estimates of the misclassification probability of the different training examples [15].

5 Conclusions

An estimator of the generalization error for bagging ensembles of arbitrary size has been developed within a Monte Carlo framework for ensemble learning. This framework allows to model the dependence of the ensemble error with smooth curves parametrized in terms of estimates of the probability that an ensemble member misclassifies a given example. The method proposed in this work computes these estimates on the out of bootstrap data, using information only from the training data. This avoids setting apart an independent dataset for validation. These estimates can be calculated efficiently, avoiding the cost of classical ensemble generalization error estimation techniques like cross validation.

Estimates of the misclassification probabilities exhibit fluctuations. This implies that error curves are biased and tend to overestimate the true error. However, this bias is shown to tend to zero as the size of the ensemble used to perform estimations grows. Experiments over several classification problems provide empirical support for the theoretical analysis of the properties of the estimator.

References

1. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
2. Esposito, R., Saitta, L.: Monte Carlo theory as an explanation of bagging and boosting. In: IJCAI, pp. 499–504. Morgan Kaufmann, San Francisco (2003)
3. Esposito, R., Saitta, L.: A Monte Carlo analysis of ensemble classification. In: Greiner, R., Schuurmans, D. (eds.) ICML, Banff, Canada, pp. 265–272. ACM Press, New York (2004)
4. Esposito, R., Saitta, L.: Experimental comparison between bagging and Monte Carlo ensemble classification. In: ICML, pp. 209–216. ACM Press, New York, USA (2005)
5. Brassard, G., Bratley, P.: Algorithmics: theory & practice. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1988)
6. Efron, B., Tibshirani, R.J.: An Introduction to the Bootstrap. Chapman & Hall/CRC (1994)
7. Quinlan, J.R.: Bagging, boosting, and C4.5. In: Proc. 13th National Conference on Artificial Intelligence, Cambridge, MA, pp. 725–730 (1996)
8. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine Learning 36(1-2), 105–139 (1999)
9. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. Journal of Artificial Intelligence Research 11, 169–198 (1999)
10. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
11. Wolpert, D.H., Macready, W.G.: An efficient method to estimate bagging's generalization error. Machine Learning 35(1), 41–55 (1999)
12. Breiman, L.: Out-of-bag estimation. Technical report, Statistics Department, University of California (1996)
13. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)
14. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall, New York (1984)
15. Nadeau, C., Bengio, Y.: Inference for the generalization error. Machine Learning 52(3), 239–281 (2003)

An Edit Distance Approach to Shallow Semantic Labeling

Samuel W.K. Chan

Dept. of Decision Sciences
The Chinese University of Hong Kong
Hong Kong SAR, China
swkchan@cuhk.edu.hk

Abstract. This paper proposes a model of semantic labeling based on the edit distance. The dynamic programming approach stresses on a non-exact string matching technique that takes full advantage of the underlying grammatical structure of 65,000 parse trees in a Treebank. Both part-of-speech and lexical similarity serve to identify the possible semantic labels, without miring into a pure linguistic analysis. The model described has been implemented. We also analyze the tradeoffs between the part-of-speech and lexical similarity in the semantic labeling. Experimental results for recognizing various labels in 10,000 sentences are used to justify its significances.

1 Introduction

Automatic information extraction has received a great deal of attention in the latest development of information retrieval. While a plethora of issues relating to questions of accuracy and efficiency have been thoroughly discussed, the problem of extracting meaning from natural language has scarcely been addressed. When the size and quantity of documents available on the Internet are considered, the demand for a highly efficient system that identifies the semantic meaning is clear. Case frame is one of the most important structures that are used to represent the meaning of sentences (Fillmore, 1968). One could consider a case frame to be a special, or distinguishing, form of knowledge structure about sentences. Although several criteria for recognizing case frames in sentences have been considered in the past, none of the criteria serves as a completely adequate decision procedure. Most of the studies in natural language processing (NLP) do not provide any hints on how to map input sentences into case frames automatically. As a result, both the efficiency and robustness of the techniques used in information extraction is highly in doubt when they are applied to real world applications.

Any high level language understanding process, such as semantic labeling, must involve chunking sentences into segments. Motivated by the psycholinguistic evidence which demonstrates that intonation changes or pauses would affect the language understanding processes in humans, Abney (1991) proposes the concept of text chunking as a first step in the full parsing. A typical chunk of a text is defined as consisting of a single content word surrounded by a constellation of function words, matching a fixed template. Church uses a simple model for finding base non-recursive

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 57–66, 2007.

NPs in sequence of POS tags (Church, 1988). Turning sentence chunking into a bracketing problem, Church calculates the probability of inserting both the open and close brackets between POS tags. Each chunking alternative is ranked and the best alternative is selected. Using transformation-based learning with rule-template referring to neighboring words, POS tags and chunk tags, Ramshaw & Marcus (1995) identify essentially the initial portions of non-recursive noun phrases up to the head, including determiners. These chunks are extracted from the Treebank parses, by selecting NPs that contain no nested NPs. While the above approaches have been proposed to recognize common subsequences and to produce some forms of chunked representation of an input sentence, the recognized structures do not include any recursively embedded NPs. As the result, the resultant fragments bear little resemblance to the kind of phrase structures that normally appear in our languages.

In this research, we propose a mechanism in shallow semantic labeling as well as sentence chunking by matching any input sentence with the trees in a Treebank through a two-phase feature-enhanced string matching. The objective of this research is twofold. First, a shallow but effective sentence chunking process is developed. The process is to extract all the phrases from the input sentences, without being bogged down into deep semantic parsing and understanding. Second, a novel semantic labeling technique that is based on the syntactic and semantic tags of the latest Treebank is being constructed (CKIP, 2004). One of our primary goals in this research is to design a shallow but robust mechanism which can annotate sentences using a set of semantic labels. The annotation will provide piecemeal the underlying semantic labels of the sentence. The organization of the paper is as follows. In our approach, each word in sentences has two attributes, i.e. part-of-speech (POS) and semantic classes (SC). Any input sentence is first transformed into a feature-enhanced string. A two-phase feature-enhanced string matching algorithm which is based on the edit distance is devised. Section 2 shows how the algorithm can be applied in the semantic labeling using 65,000 parse trees in a Treebank. The system has already been implemented using Java language. In order to demonstrate the capability of our system, an experiment with 10,000 sentences is conducted. A detailed evaluation is explained in Section 3 followed by a conclusion.

2 Two-Phase Feature-Enhanced String Matching Algorithm

In this section, we will first outline the concepts of edit operations which are essential components of our feature-enhanced string matching algorithm. The two-phase shallow semantic labeling will be discussed thoroughly in Section 2.2.

2.1 Edit Operations

Our algorithm is essentially accomplished by applying a series of edit operations to an input sentence to change it to every tree in the Treebank. Every edit operation has been associated with a cost and the total cost of the transformation can be calculated by summing up the costs of all the operations. This cost reflects the dissimilarity between the input sentence and the trees. Instead of analyzing the exact words in the sentence, extended attributes of each word in both input sentence and the trees, with their POS and semantic classes, are used. The closely matched tree, i.e., the

one with minimum cost, or called *edit distance*, is selected and the corresponding phrase structures and semantic labels delineated in the tree are unified with the input sentence.

Let two given feature-enhanced strings A and B denoted as $A = a_1a_2a_3... a_m$ and $B = b_1b_2b_3... b_n$, where are a_i, b_j the ith and jth attributed symbols of A and B respectively. Each attributed symbol represents a primitive of A or B. Generally speaking, to match a feature-enhanced string A with another B means to transform or edit the symbols in A into those in B with a minimum-cost sequence of allowable edit operations. In general, the following three types of edit operations are available for attributed symbol transformation.

(a) *Change*: to replace an attributed symbol a_i with another b_j, denoted as $a_i \rightarrow b_j$.
(b) *Insert*: to insert an attributed symbol b_j into a feature-enhanced string, denoted as $\lambda \rightarrow b_j$ where λ denotes a null string.
(c) *Delete*: to delete an attributed symbol a_i from a feature-enhanced string, denoted as $a_i \rightarrow \lambda$.

Definition 1. An *edit sequence* is a sequence of ordered edit operations, $s_1, s_2,... s_p$ where s_i is any of the following three types of edit operations, *Change*, *Insert*, *Delete*.

Definition 2. Let R be an arbitrary nonnegative real cost function which defines a cost $R(a_i \rightarrow b_j)$ for each edit operation $a_i \rightarrow b_j$. The cost of an edit sequence $S = s_1, s_2,... s_p$ to be

$$R(S) = \sum_{i=1}^{p} R(s_i) \tag{1}$$

Definition 3. For two strings A ad B with length m and n respectively, $D(i, j)$ denotes the edit distance, which is the minimum number of edit operations, needed to transform the first i characters of A into first j characters of B, where $1 \le i \le m$ and $1 \le j \le n$.

In other words, if A has m letters and B has n letters, then the edit distance of A and B is precisely the value $D(m, n)$. Wagner & Fischer (1974) had proposed the following algorithm for computing every edit distances $D(i, j)$.

```
[Algorithm]
D(0, 0) := 0;
for i := 1 to len(A) do D(i, 0):=D(i-1, 0)+R(a_i→λ);
for j := 1 to len(B) do D(0, j):=D(0, j-1)+R(λ→b_j);
for i := 1 to len(A) do
    for j := 1 to len(B) do
    begin
          m1 := D(i, j-1) + R(λ→b_j);
          m2 := D(i-1, j) + R(a_i→λ);
          m3 := D(i-1, j-1) + R(a_i→b_j);
          D(i, j) := min (m1, m2, m3);
    end
```

Our feature-enhanced string matching in shallow semantic labeling is to make use of the algorithm above and modify the cost function $R(.)$ for various edit operations.

2.2 Shallow Semantic Labeling as Two-Phase Feature-Enhanced String Matching

Our labeling is defined as a two-phase feature-enhanced string matching using the edit operations. For every input sentence, a coarse-grained syntactic matching is conducted in our first phase of matching. The matching relies on a set of coarse-grained but global part-of-speech (POS) tags. The major objective of this phase is to shortlist all the potential trees among 65,000 parse trees in the CKIP Treebank, which are relevant to the input sentence, without getting bogged down into computational complexity with other linguistic details. The second phase of the matching is followed to compute the dissimilarity measure between the input sentence and every short-listed candidate that is identified in the first phase. Detailed POS and semantic class (SC) tags will be employed. As a result, a candidate tree which has the minimum dissimilarity with the input sentence will be identified. The underlying semantic labels and phrases of the candidate tree are used to determine the shallow language patterns of the input sentence. The details of the two-phase matching are explained in the folloing.

2.2.1 Coarse-Grained Syntactic Matching

In the first phase of matching, each word is represented by its corresponding POS. Let S be an input sentence and the T be a tree in a Treebank, s_i and t_j be two tokens in S and T with attribute POS_i and POS_j respectively. We define the cost function for the *change* operation $s_i \rightarrow t_j$ to be

$$R(s_i \rightarrow t_j) = u(POS_i, POS_j) \tag{2}$$

where $u(POS_i, POS_j)$ defines the cost due to the difference between the POS of the two tokens. The POS tags from the Chinese Knowledge Information Processing Group (CKIP) of Academia Sinica are employed (Chen *et al.*, 1996). The tags are subdivided into 46 major POS classes which are further refined into more than 150 subtypes. However, in this coarse-grained matching, only the major POS classes will be considered. To figure out the cost function $u(\cdot,\cdot)$ in the coarse-grained matching, all the major POS tags are organized into a hierarchical structure with an associated hard-coded cost function. Figure 1 shows the structure of notional words and describes the relative distances between the adjectives (`A`), verbs (`V`), status-verbs (`VH`), measure-words (`Nf`), nouns (`N`), position-words (`Ng`), time-words (`Nd`) and place-words (`Nc`). All notional words have definite meanings in the language. The cost function is based on their interchangeability, the degree of flexibility in placement in the syntax, and the similarity of their acceptable modifiers. For example, Chinese verbs and adjectives share a lot of common features syntactically, i.e. both can be predicates or modified by adverbs and the word, *not*. All these features fail to appear in nouns. The abbreviations in bracket indicate the original POS tags marked by the CKIP. The corresponding tree structure of the XML is shown in Figure 2. The cost function $u(\cdot,\cdot)$ reflects the difference based on the tag `toll` encoded in the XML as shown in Figure 1. The function also indicates the degree of alignment between the syntactic

structure of the input sentence and the trees in the Treebank. Although two feature-enhanced strings with the same POS sequence do not imply they will share the same syntactic structure, this coarse-grained syntactic matching shortlists the potential trees by imposing a necessary, even not sufficient, constraint on its syntactic structure and limits the potential search space in the subsequent stage of semantic matching.

```
<Head toll="5">
  <NodeB toll="2">
    <NodeC toll="2">
      <Adjective toll="5"/>
      <Verb toll="5"/>
    </NodeC>
    <Status-Verb toll="7"/>
  </NodeB>
  <NodeD toll="2">
    <Measure-Word toll="7"/>
    <NodeE toll="2">
      <Noun toll="5"/>
      <NodeF toll="2">
        <Position-word toll="3"/>
        <NodeG toll="1">
          <Time-word toll="2"/>
          ...
</Head>
```

Fig. 1. XML illustrating the relative distances between 8 different types of POS

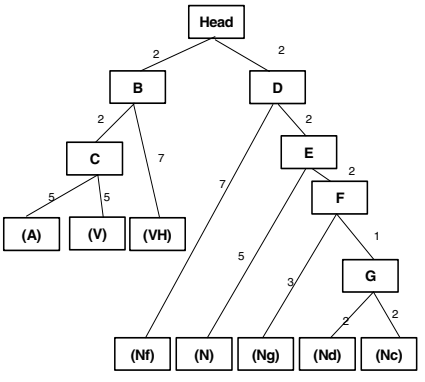


Fig. 2. Corresponding tree structure of the XML shown in Fig.1

2.2.2 Computation of Semantic Dissimilarity

What this second phase matching basically does is to make a detailed comparison between the input sentence and the short-listed trees in its earlier stage. In this phase, each Chinese token has two attributes, i.e. a detailed part-of-speech (*POS*) and semantic class (*SC*). Similar to the approach in Section 2.2.1, we define the cost function for the *change* operation $s_i \rightarrow t_j$ to be

$$R(s_i \rightarrow t_j) = f\left(u(POS_i, POS_j), v(SC_i, SC_j)\right) \tag{3}$$

where the function *f* is the dissimilarity function relied on two major components. The first component $u(POS_i, POS_j)$ defines the partial cost due to the difference between the detailed POS of the words. The detailed POS tags are organized in XML format, similar to the approach demonstrated in Figure 1. For example, the further breakdown of the nouns (`Na`) which are divided into in-collective (`Nae`) and collective (`Na1`) nouns. The collective nouns are then subdivided into in-collective concrete uncountable nouns (`Naa`), in-collective concrete countable nouns (`Nab`), in-collective abstract countable nouns (`Nac`), and in-collective abstract uncountable nouns (`Nad`). The second term in Eqn. (3) defines another partial cost due to the semantic differences. In our approach, the words in the input sentences and the trees are identified using a bilingual thesaurus similar to the Roget's Thesaurus. The *is-a* hierarchy in the bilingual thesaurus, shown the underlying ontology, can be viewed as a directed acyclic graph with a single root. While the upward links correspond to generalization, the specialization is represented in the downward links. The ontology demonstrated in the

thesaurus is based on the idea that linguists classify lexical items in terms of similarities and differences. They are used to structure or rank lexical items from more general to the more special. Based on the *is-a* hierarchy in the thesaurus, we define the conceptual distance d between two notional words by their shortest path lengths. Given two words t_1 and t_2 in an *is-a* hierarchy of the thesaurus, the semantic distance d between the tokens is defined as follows:

$$d(t_1, t_2) = \text{minimal number of } is\text{-}a \text{ relationships in the shortest path between } t_1 \text{ and } t_2 \tag{4}$$

The shortest path lengths in *is-a* hierarchies are calculated. Initially, a search fans out through the *is-a* relationships from the original two nodes to all nodes pointed to by the originals, until a point of intersection is found. The paths from the original two nodes are concatenated to form a continuous path, which must be a shortest path between the originals. The number of links in the shortest path is counted. Since $d(t_1, t_2)$ is positive and symmetric, $d(t_1, t_2)$ is a metric which means (i) $d(t_1, t_1) = 0$; (ii) $d(t_1, t_2) = d(t_2, t_1)$; (iii) $d(t_1, t_2) + d(t_2, t_3) \geq d(t_1, t_3)$. At the same time, the semantic similarity measure between the items is defined by:

$$v(t_i, t_j) := \begin{cases} d(t_i, t_j) & \text{if } d(t_i, t_j) \leq d_{max} \\ MaxInt & \text{otherwise} \end{cases} \tag{5}$$

where d_{max} is proportional to the number of lexical items in the system and *MaxInt* is a maximum integer of the system. This semantic similarity measure defines the degree of relatedness between the words. Obviously, strong degree of relatedness exists between the lexical tokens under the same nodes. On the other hand, for the cost of the *insert* and *delete* operations, we make use the concept of *collocation* that measures how likely two words are to co-occur in a window of text. To better distinguish statistics based ratios, work in this area is often presented in terms of the pointwise mutual information (*MI*), which is defined as

$$MI(t_{j-1}, t_j) = \log \frac{P(t_{j-1}, t_j)}{P(t_{j-1}) \times P(t_j)} \tag{6}$$

where t_{j-1} and t_j are two adjacent words (Manning & Schütze, 1999). While $P(t_{j-1}, t_j)$ is the probability of observing t_{j-1} and t_j together, $P(t_{j-1})$ and $P(t_j)$ are the probabilities of observing t_{j-1} and t_j anywhere in the text, whether individually or in conjunction. Note that tokens that have no association with each other and co-occur together according to chance will have a *MI* value close to zero. This leads to the cost function for insertion and deletion shown in Eqns. (7) and (8) respectively.

$$R(\lambda \rightarrow t_j) = \begin{cases} k \times e^{-z} & \text{if } z > \varepsilon > 0 \\ MaxInt & \text{otherwise} \end{cases} \tag{7}$$

where $z = \min \{MI(t_{j-1}, t_j), MI(t_j, t_{j+1})\}$

$$R(t_j \rightarrow \lambda) = \begin{cases} l \times e^{-MI(t_{j-1}, t_{j+1})} & \text{if } MI(t_{j-1}, t_{j+1}) > \varepsilon > 0 \\ MaxInt & \text{otherwise} \end{cases} \tag{8}$$

where k, l, ε are three constants relied on the size of the active corpus.

Obviously, the insertion operation will be penalized if the co-occurrence between the newly inserted word and its neighbors is low. Similarly, the deletion operation is most likely to happen if there is a high co-occurrence between the adjacent pairs after the deletion. Using the above cost functions for the three types of edit operations, the tree in the Treebank with minimum cost is being chosen to be the best approximation of the input sentence and its associated semantic labels will be adopted. Shallow language patterns are then extracted based on the recursive structures and semantic labels appeared in the Treebank. The experimental results of the semantic labeling are shown in the section below.

3 Experimental Results

As mentioned in Eqn. (3), several approaches have been used to define the dissimilarity function f by combining the semantic differences and the detailed POS tags in our second phase feature-enhanced string matching. In our evaluations, five different types of dissimilarity function f are applied. They are

(i) $f_1(u, v) = u(POS_i, POS_j)$
(ii) $f_2(u, v) = v(SC_i, SC_j)$
(iii) $f_3(u, v) = u(POS_i, POS_j) + v(SC_i, SC_j)$
(iv) $f_4(u, v) = \max\,(u(POS_i, POS_j)\ ,\ v(SC_i, SC_j))$

Dissimilarity function $f_1(u, v)$ provides a detailed version of our coarse-grained syntactic matching. Detailed POS tags are used as the dissimilarity measure in the labeling. Similarly, $f_2(u, v)$ considers only the semantic class of the words. The other two combine both syntactic and semantic features in defining the dissimilarity measures. We have tested our shallow semantic labeling with 10,000 sentences with the Treebank. Since this research is concerning with shallow semantic labeling, we have no incentive to match the trees/subtrees in the Treebank with very complicated structures. The average sentence length is around 13.7 characters per sentence.

Table 1. Sentence analysis in the experiment. Edit distance is defined as a minimum cost in transforming the input sentence with the closest sentence pattern in the Treebank.

Dissimilarity function f	Range of Edit distance	% of sentences	Average edit distance	% of sentences w/ incomplete info.
$f_1(u, v)$	0-25	13.9	19.2	2.1
	26-50	16.3	40.5	2.1
	51-75	19.7	63.6	4.7
$f_2(u, v)$	0-25	11.3	19.3	2.3
	26-50	15.6	41.4	3.8
	51-75	17.7	65.2	6.4
$f_3(u, v)$	0-25	24.1	17.9	2.2
	26-50	31.6	38.2	4.1
	51-75	22.7	62.3	6.9
$f_4(u, v)$	0-25	20.5	19.6	1.9
	26-50	22.4	40.9	3.8
	51-75	26.9	58.2	7.3

Table 1 summarizes the results of our system evaluation. The third and fourth columns in the table are number of sentences in each range of edit distance and their average edit distances. The edit distance is defined as a minimum cost in transforming the input sentence with the closest sentence pattern in the Treebank. In other words, the smaller the distance, the higher similarity they have. If it is considered as a good match where the edit distances are equal to or less than 50, then it can be observed, in Table 1, that the dissimilarity functions f_3 and f_4 all produce higher percentage of sentences with lower edit distance. This reflects both the information from syntactic tags and semantic classes provide useful clues in our shallow semantic labeling. Our experiments are not conducted with perfect information. It is worthwhile to mention that, as shown in right-most column of Table 1, more than 530 sentences have incomplete information which mainly comes from proper nouns, or out-of-vocabulary (OOV) words. Both of them have neither defined POS nor semantic class. All these information will be annotated with a default value which will certainly induce errors in our labeling. While it is inevitable to have OOV words in any real corpus, the performance, due to the coverage of POS and semantic classes, does not deteriorate much in our system. The labeling is still feasible over the sentences with OOV words. This tolerance ability provides the graceful degradation in our shallow semantic labeling. While other systems are brittle and working only in all-or-none basis, the robustness of our system is guaranteed. At the same time, while real text tends to have grammatical mistakes and error-prone, these problems can be tackled with an acceptable tolerance in our system. In our second evaluation, we have tested our algorithm in recognizing several major semantic labels that appear in our sentences. The semantic labels include `theme`, `goal`, `property`, `range`, `agent`, and `predication`. As with other text analysis, the effectiveness of the system appears to be dictated by recall and precision parameters where recall (R) is a percentage of how many correct labels can be identified while precision (P) is the percentage of labels, tackled by our system, which are actually correct. In addition, a common parameter F is used as a single-figure measure of performance which combines recall (R) and precision (P) as in follows,

$$F = \frac{(\beta^2 + 1) \times P \times R}{\beta^2 \times P + R} \tag{9}$$

We set $\beta = 1$ to give no special preference to either recall or precision. The recall, precision and F-score for the semantic labels in dissimilarity function f_3 are shown in Table 2.

Table 2. Evaluation of some semantic labels in the dissimilarity function f_3. An elementary subtree spans only on a sequence of words while a derivation subtree contains at least one branch of elementary subtree.

	Elementary Subtree			**Derivation Subtree**		
Semantic Label	***R***	***P***	***F*-score**	***R***	***P***	***F*-score**
`theme`	0.79	0.82	0.805	0.88	0.85	0.865
`goal`	0.80	0.79	0.795	0.78	0.76	0.770
`property`	0.89	0.91	0.900	0.78	0.83	0.804
`range`	0.94	0.92	0.930	0.93	0.91	0.920
`agent`	0.92	0.87	0.894	0.92	0.85	0.884
`predication`	0.76	0.81	0.784	0.80	0.78	0.790
`all labels`	0.81	0.83	0.821	0.78	0.81	0.801

As shown in the last row in Table 2, the precision and recall of all semantic labels are calculated by considering all the semantic labels that appear in the sentences, rather than by averaging the measures for individual semantic labels. It is worth noting that the greatest differences in performance are the recall while the precision remains relatively steady in most semantic labels. One possible explanation is that the low recall rates in some labels are due to less complete coverage of linguistic phenomena. In addition, we define an elementary subtree that spans only on a sequence of words, as well as a derivation subtree that contains at least one branch of elementary subtree. It may be expected the *F*-score of the derivation subtrees will be much worse than its counterpart, however, Table 2 shows surprisingly the differences in the overall accuracy in two main types of subtrees are not significant. An explanation is that we have approached chunking as well as assigning the most salient semantic label to the chunks based on the POS and semantic tags. Even though there may be some misclassification in the terminal nodes, this will not hinder the system to tag the semantic labels in the longer chunks. In other words, the longer chunks are less error prone in our semantic labeling. This shallow semantic labeling technique produces an output that abstract away the details but retains the core semantic structure of the actual sentence. To further evaluate our system, we compare its performance with other approaches, even though the relevant experimental results in Chinese language are not easy to obtain in the literature. We compare our system with those systems participated in the CoNLL, the Conference on Natural Language Learning. The training data used in the CoNLL-2005 consists of sections from the Wall Street Journal with information on predicate-argument structures (Carreras & Màrquez, 2005). Table 3 shows the performance comparison with the top five participating systems in the conference. Even though our system cannot compare head-to-head with their models since their training and test data is totally disparate with our experiment, this paper has suggested one of the alternatives in semantic labeling, with *F*-score over 0.8 shown in Table 2. Certainly, further explorations are needed to improve the system performance.

Table 3. Overall precision (*P*), recall (*R*) and *F*-scores of the top five systems in CoNLL-2005. While the second column, ML-Method, illustrates the related machine learning techniques, the third column shows whether they involve any post-processing method.

System	ML-Method	Post	*P*	*R*	*F*
Punyakanok *et al* (2005)	Winnow-based Network	No	0.823	0.768	0.794
Haghighi *et al* (2005)	Maximum Entropy	No	0.795	0.774	0.785
Marquez *et al* (2005)	AdaBoost Algorithm	No	0.795	0.765	0.780
Pradhan *et al* (2005)	Support Vector Machines	No	0.819	0.733	0.774
Surdeanu & Turmo (2005)	AdaBoost Algorithm	Yes	0.803	0.730	0.765

5 Conclusion

We have illustrated a shallow technique in which semantic labels are extracted in forms of chunks of phrases or words using a two-phase feature-enhanced string matching algorithm. This shallow technique is inspired by the research in the area of bio-molecular sequences analysis which advocates high sequence similarity usually implies significant function or structural similarity. It is characteristic of biological systems that objects have a certain form that has arisen by evolution from related

objects of similar but not identical form. This sequence-to-structure mapping is a tractable, though partly heuristic, way to search for functional or structural universality in biological systems. With the support from the results as shown, we conjecture this sequence-to-structure phenomenon appears in our sentences. The sentence sequence encodes and reflects the more complex linguistic structures and mechanisms described by linguists. While our system does not claim to deal with all aspects of language, we suggest an alternate, but plausible, way to handle the real corpus.

Acknowledgments. The work described in this paper was partially supported by the grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. CUHK4438/04H and CUHK4706/05H).

References

Abney, S.: Parsing by chunks. In: Berwick, R., Abney, S., Tenny, C. (eds.) Principle-Based Parsing, Kluwer Academic, Dordrecht (1991)

Carreras, X., Màrquez, L.: Introduction to the CoNLL-2005 shared task: Semantic role labeling. In: CoNLL. Proceedings of the 9th Conference on Computational Natural Language Learning, pp. 152–164 (2005)

Chen, K.-J., Huang, C.-R., Chang, L.-P., Hsu, H.-L.: Sinica Corpus: Design Methodology for Balanced Corpora. In: PACLIC II. Proceedings of the 11th Pacific Asia Conference on Language, Information, and Computation, Seoul Korea, pp. 167–176 (1996)

Church, K.: A stochastic parts program and noun phrase parser for unrestricted text. In: Proceedings of Second Conference on Applied Natural Language Processing, Austin, Texas (1988)

CKIP. Sinica Chinese Treebank: An Introduction of Design Methodology. Academic Sinica (2004)

Fillmore, C.J.: The case for case. In: Bach, E., Harms, R.T. (eds.) Universals in Linguistic Theory, pp. 1–90. Rinehart & Winston, Holt (1968)

Haghighi, A., Toutanova, K., Manning, C.: A joint model for semantic role labeling. In: CoNLL. Proceedings of the 9th Conference on Computational Natural Language Learning (2005)

Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)

Màrquez, L., Comas, P.R., Giménez, J., Català, N.: Semantic role labeling as sequential tagging. In: CoNLL. Proceedings of the 9th Conference on Computational Natural Language Learning (2005)

Pradhan, S., Hacioglu, K., Ward, W., Martin, J.H., Jurafsky, D.: Semantic role chunking combining complementary syntactic views. In: CoNLL. Proceedings of the 9th Conference on Computational Natural Language Learning (2005)

Punyakanok, V., Koomen, P., Roth, D., Yih, W.: Generalized inference with multiple semantic role labeling systems. In: CoNLL. Proceedings of the 9th Conference on Computational Natural Language Learning (2005)

Ramshaw, L.A., Marcus, M.P.: Text chunking using transformation-based learning. In: Proceedings of the Third Workshop on Very Large Corpora, pp. 82–94 (1995)

Surdeanu, M., Turmo, J.: Semantic role labeling using complete syntactic analysis. In: CoNLL. Proceedings of the 9th Conference on Computational Natural Language Learning (2005)

Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. Journal of the Association for Computing Machinery 21(1), 168–173 (1974)

A Comparison of One-Class Classifiers for Novelty Detection in Forensic Case Data*

Frédéric Ratle[1], Mikhail Kanevski[1], Anne-Laure Terrettaz-Zufferey[2], Pierre Esseiva[2], and Olivier Ribaux[2]

[1] Institute of Geomatics and Risk Analysis, Faculty of Earth and Environmental Sciences, University of Lausanne, CH-1015, Switzerland
frederic.ratle@unil.ch

[2] School of Criminal Sciences, Faculty of Law, University of Lausanne, CH-1015, Switzerland

Abstract. This paper investigates the application of novelty detection techniques to the problem of drug profiling in forensic science. Numerous one-class classifiers are tried out, from the simple k-means to the more elaborate Support Vector Data Description algorithm. The target application is the classification of illicit drugs samples as part of an existing trafficking network or as a new cluster. A unique chemical database of heroin and cocaine seizures is available and allows assessing the methods. Evaluation is done using the area under the ROC curve of the classifiers. Gaussian mixture models and the SVDD method are trained both with and without outlier examples, and it is found that providing outliers during training improves in some cases the classification performance. Finally, combination schemes of classifiers are also tried out. Results highlight methods that may guide the profiling methodology used in forensic analysis.

1 Introduction

Analytical techniques such as gas chromatography are becoming widespread in forensic science in order to find underlying patterns in crime-related data, especially in the analysis of illicit drugs composition. Indeed, it has become largely accepted that the chemical signature of drug samples can provide information about the origin or the distribution network of the products and producers. An important issue that arises in this application is, given a set of chemical samples which can be related to known criminal investigations, how can one characterize this dataset in order to determine if a new sample can be linked to a known data cluster. If it cannot, it could be part of a "new" cluster. To this end, one-class classification is a novel and efficient way of approaching this problem.

In this paper, we perform a comparison of several popular one-class classifiers to the problem of drug profiling. The aim is to determine the most promising

* This work is supported by the Swiss National Science Foundation (grant no.105211-107862).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 67–76, 2007.

methods for this application, and to find potential strengths and weaknesses of the novelty detectors. A remarkable characteristic of the datasets is that class labels corresponding to links confirmed by investigators are available and allow a real evaluation of the performance of the methods and of the relevance of the chemical composition of drugs in order to classify samples.

2 Related Work

Introductory work on chemical drug profiling in forensic science can be found in [1] and [2]. In these papers, no "true" class labeling is available; only chemical similarities are used as class membership criteria. Nonetheless, a profiling method based on samples correlation is devised. Several distance measures are used, and results are good when considering only chemical links as class criteria.

The datasets used here have been previously studied by the present authors in [3] and [4] using nonlinear dimensionality reduction techniques and various classification algorithms. In [5], authors apply the SVDD algorithm to novelty detection in mass spectra data. However, since no class labels are available, the performance of SVDD is assessed using a comparison with a clustering method.

3 Novelty Detection

Novelty detection, also called one-class classification, is usually defined as the task of detecting a signal or pattern that a learning system is not aware of during training. Broad reviews of the subject can be found in [6], [7] and [8]. Even though the problem of outlier detection is a classical one in statistics, one-class classifiers have only been popularized recently. Most statistical approaches, such as Mahalanobis distance or extreme value theory, rely on strong assumptions, which are not always respected when dealing with small and noisy datasets. Many machine learning approaches, apart from density-based methods, go round these assumptions by trying to model the support of the data rather than its whole distribution. As suggested by Tax [9], one-class classifiers usually fall into one of these categories: density estimation methods, boundary methods and reconstruction methods.

3.1 Density Estimation Methods

Density estimation methods aim at estimating the whole distribution of the target data. A rejection threshold is then fixed so that points located in the far tails of the distribution are rejected. We shortly describe here the three density-based methods used in this study.

Gaussian Distribution. Here, a single Gaussian distribution is fitted to the target data. The mean and covariance matrix is estimated from the data, and the points comprised in the two tails are considered outliers. The rejection threshold is set such that 5% of the target data is rejected.

Parzen Density Estimation. The Parzen density estimator is a mixture of kernels - typically Gaussian - with each of them centered on one training point. It can be expressed simply as

$$p(\mathbf{x}) = \sum_{i=1}^{N} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) \tag{1}$$

N is the size of the target training set and K is the kernel. One parameter, the width h (smoothing parameter), has to be tuned. Again, the rejection threshold is set such that 5% of the target data is rejected.

Gaussian Mixture Models. Gaussian mixture models (GMM) are used to characterize the target data distribution by using a linear combinations of Gaussian distributions. Generally speaking, the likelihood of a mixture of Gaussians can be expressed as

$$p(\mathbf{x}) = \sum_{i=1}^{K} \pi_k N(\mathbf{x}|\mu_k, \mathbf{\Sigma}_k) \tag{2}$$

The π_k's are scalar weights. Unlike Parzen density estimation, the number of Gaussians is specified and usually much smaller than the size of the training set. The means μ_k and covariances $\mathbf{\Sigma}_k$ are estimated by maximum likelihood using the expectation-maximization (EM) algorithm.

Two variants of GMM are tested: Gaussians with a diagonal covariance matrix and with a full covariance matrix. In the former case, elliptic clusters are assumed, while the latter case can take into account arbitrary-shaped clusters.

3.2 Boundary Methods

Boundary methods, rather than estimating the distribution, aim at constructing a boundary - such as a sphere - around the target data. Points that fall outside the limits of the boundary are rejected. Here, k-nearest neighbors and SVDD are used.

K-Nearest Neighbors. KNN first calculates the distances of the test point to its k neighbors, and averages these distances in order to have a single measure. It then computes the distances from these k neighbors to *their* k-nearest neighbors. Based on these distances, the local density of each point is computed, and the new point is rejected if its local density is inferior to that of its neighbors in the training set.

Support Vector Data Description. Support vector data description (SVDD), introduced in [10], is a method for characterizing the target data distribution without explicitely estimating the distribution parameters. It works by fitting the smallest possible hypersphere around the target data in the feature space induced by a specified kernel, typically a Gaussian kernel. Data points that fall outside the hypersphere when projected in the feature space are rejected.

This method has many similarities with the support vector method for novelty detection presented in [11]. However, in the latter work, an optimal hyperplane is built between target and outlier data, while SVDD builds a hypersphere.

3.3 Reconstruction Methods

The goal of reconstruction methods is to develop a simplified representation of the data via clusters or principal components. These methods are numerous: k-means, principal components analysis, self-organizing maps, etc. Only k-means has been chosen among the reconstruction methods.

K-means. In order to perform k-means clustering, the number of clusters in the target data has to be specified. Following this, boundaries are constructed around each cluster such that a certain fraction (5% here) of the target data is rejected. Again, points that fall outside the boundaries are considered outliers. It can be supposed that this type of method will work best for clusters that are well-separated.

3.4 Combination of Classifiers

Ensemble methods have become increasingly popular when dealing with noisy real-world problems. This is also true for the problem of one-class classification, for which combination schemes have been proposed [12]. In this paper, we test two approaches: average and product of the posterior probabilities of the classifiers. These probabilities are either directly obtained when using a density-based method, or estimated when using reconstruction or boundary methods.

3.5 ROC Analysis

A very useful assessment tool in classification and novelty detection tasks is the well-known receiver operating characteristic (ROC) curve. This curve represents the true positives (targets accepted as such) plotted against the false positives (outliers accepted as target), when varying the acceptation threshold. The area under the ROC curve (AUC) is thus a good measure of the classification performance. A random guess classifier is expected to have an AUC of 0.5, if the number of samples is large enough, while a classifier achieving a perfect separation will have an AUC of 1. Consequently, the AUC criterion must be maximized in order to obtain a good separation between targets and outliers.

4 Datasets and Methodology

4.1 The Data

Many types of substances can be found in a drug sample, and each of these can possibly provide information about a certain stage of drug processing. The

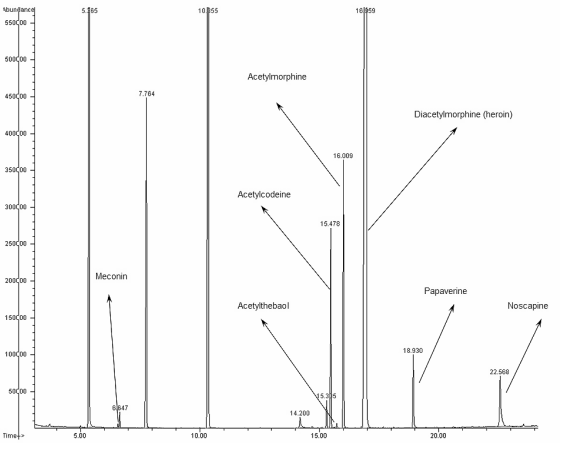


Fig. 1. An example of a chromatogram for a heroin sample. Each feature corresponds to the proportion of a constituent, estimated by the area under its corresponding peak.

interested reader may find a thorough description of this processing in [13]. This study focusses on the major chemical constituents, measured using GC/FID (gas chromatography and flame ionization detector). Details regarding the experimental procedure can be found in [1]. Each sample is characterized by features corresponding to the proportion of each chemical it contains. The first dataset (heroin) has 7 features, while the second (cocaine) has 13 features. The proportions of the chemical constituents have been estimated for each sample by using the area under the peaks in its chromatogram, after removal of the background noise. Figure 1 shows a typical chromatogram of a heroin sample.

Fig. 2 shows the labeled datasets projected on their two first principal components, in order to give an indication of the type of clusters that might be encountered. These figures show that the classes vary in shape and exhibit different scales. This could be expected, since the class labeling corresponds to networks of people involved in trafficking, while the input data corresponds to chemical constituents. It is thus of no surprise that the correlation between chemical profiles may not always match the links found by investigation, since two persons linked within a network do not necessarily share identical products from a chemical perspective.

The data consist of 323 heroin samples (with originally 3 classes) and 310 cocaine samples (10 classes). Each class corresponding to a distinct case (regardless of the chemical content of the samples), we have drawn out and tagged as outliers one case from the heroin dataset and two cases from the cocaine dataset, the number of classes being superior for the latter. Each dataset thus contains 1 target class (containing the remaining original classes) and 1 outlier class (the drawn out cases). The rationale for this is that we want to classify a sample as being linked to a known network or not.

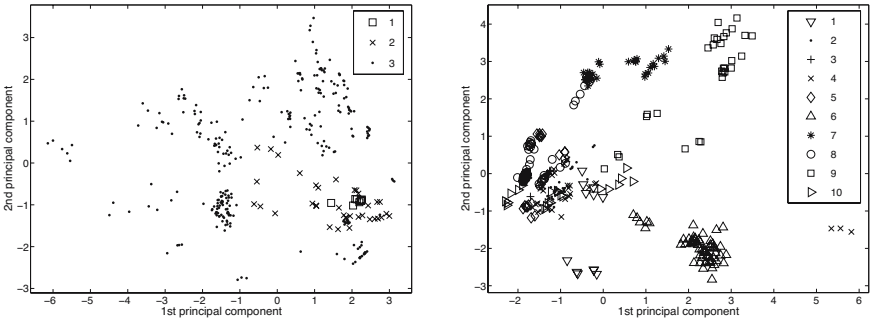


Fig. 2. Projection on the two first principal components of heroin (left) and cocaine (right)

4.2 Experimental Setup

All experiments have been performed in Matlab. The Data Description toolbox (DDTools) [14] has been used. When necessary, parameter values (σ or k) were assigned using line search and a k-fold cross-validation scheme. The main difference with normal k-fold cross-validation is that, outliers being available, the outlier set was also split into k folds, but not used for training. The procedure can be summarized as follows:

1: **for** $i = 1$ to k **do**
2: Remove partition p_i^T of target dataset T to obtain T'.
3: Remove partition p_i^O of outlier dataset O.
4: Train the one-class classifier on T'.
5: Compute the AUC a_i for dataset $p_i^T \cup p_i^O$.
6: **end for**
7: Compute the cross-validation error $e = \frac{1}{K} \sum_{j=1}^{K} a_i$.

As the datasets are rather small given the number of variables, using only half of the training targets to test the classifiers may not allow a good characterization of the target data. Two measures are thus given:

1. AUC on training target data and independent test outliers (called training AUC or simply AUC below).
2. AUC by the k-fold cross-validation method previously described (using 5 folds, and called AUC-CV).

The first method obviously overestimates the AUC, while the second might be both pessimistic or optimistic. However, our prime goal here is to compare the methods one against another.

5 Results and Discussion

Tables 1 and 2 show the obtained results, which are averaged over 10 runs.

Table 1. AUC for the heroin and cocaine dataset, without outliers in the training process. GMM I designates the Gaussian mixture model with a diagonal covariance matrix, and GMM II the model with the full matrix.

	Heroin			Cocaine		
	AUC	AUC-CV	param.	AUC	AUC-CV	param.
K-means	64.6 ± 0.0	51.5 ± 4.5	$k = 2$	95.4 ± 5.3	84.9 ± 4.1	$k = 8$
KNN	97.7 ± 0.0	62.2 ± 0.0	$k = 2$	96.6 ± 0.0	87.7 ± 0.0	$k = 9$
Gauss	64.0 ± 0.0	45.7 ± 0.0	-	98.4 ± 0.0	90.8 ± 0	-
Parzen	92.1 ± 0.0	58.7 ± 0.0	$h = 0.5$	94.8 ± 0.0	89.0 ± 0.0	$h = 1.5$
GMM I	55.3 ± 0.2	41.5 ± 8.7	2 clusters	87.7 ± 0.0	86.4 ± 2.2	8 clusters
GMM II	84.7 ± 3.9	62.5 ± 7.1	2 clusters	98.1 ± 1.5	83.2 ± 2.1	8 clusters
SVDD	59.6 ± 12.0	66.4 ± 7.8	$\sigma = 2$	88.6 ± 3.6	90.2 ± 1.1	$\sigma = 3$
all-mean	91.0 ± 0.8	60.0 ± 1.2	-	98.2 ± 1.2	87.2 ± 2.2	-
all-product	91.1 ± 0.7	60.1 ± 1.3	-	99.1 ± 0.8	87.2 ± 2.2	-

Table 2. AUC for the heroin and cocaine dataset, with outliers in the training process

	Heroin			Cocaine		
	AUC	AUC-CV	param.	AUC	AUC-CV	param.
GMM I	70.9 ± 9.4	42.7 ± 8.3	2 clusters	98.3 ± 0.8	86.0 ± 3.0	8 clusters
GMM II	84.8 ± 3.0	64.8 ± 7.1	2 clusters	99.8 ± 0.1	84.7 ± 2.0	8 clusters
SVDD	78.4 ± 2.4	40.5 ± 4.1	$\sigma = 2$	96.2 ± 0.3	86.7 ± 1.5	$\sigma = 3$
all-mean	83.3 ± 4.6	60.2 ± 4.1	-	98.6 ± 0.4	86.0 ± 1.9	-
all-product	82.2 ± 2.1	57.2 ± 7.7	-	99.7 ± 0.2	85.4 ± 1.9	-

Figure 3 shows ROC curves on test outliers and training target data for the best and the worst classifier. Since the test AUC was estimated with cross-validation, the corresponding curves cannot be illustrated. It can be seen that the performances for the second dataset are located within a smaller interval.

Results show a surprising difference between the two datasets regarding the general performance of the methods. First, for the heroin dataset, the gap between AUC and AUC-CV is considerably larger than that of the cocaine dataset. KNN and Parzen perform best at achieving a good separation between target and outlier data, but they both provide an average performance on cross-validation. The GMM with a full covariance matrix has produced above average results for both AUC and AUC-CV, while the GMM with a diagonal matrix has performed poorly. The SVDD method, even though providing a poor training AUC, seems to be by far the most robust. The AUC has not decreased at all between the training AUC and AUC-CV. In fact, the performance improved, but this can probably be explained by an "optimistic" partitioning during cross-validation. K-means and the Gaussian distribution performed below average. The latter even performed worse than random guessing on AUC-CV, as did GMM I.

On the cocaine dataset, most methods are more consistent. Indeed, the AUC-CV is much more closer to the training AUC for all the methods. All algorithms performed reasonably well, although SVDD has again appeared slightly more

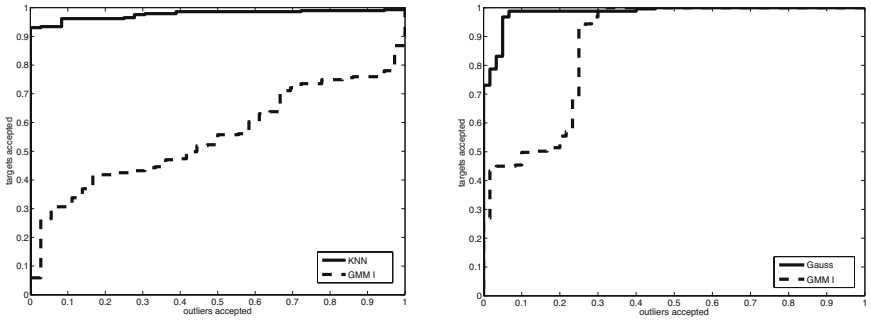


Fig. 3. ROC curves for the best and the worst classifiers obtained on training target data and independent outliers, for both datasets (heroin on the left and cocaine on the right). Since the test AUC was estimated using cross-validation, the corresponding ROC curves cannot be obtained. For the second dataset, the classification performance is comprised within a smaller interval.

robust. Some methods have improved significantly when applied to this dataset. While the Gaussian distribution was among the worst classifiers for heroin data, it outperforms all the other methods on cocaine data. Some of these observations are summarized in Table 3.

Results of classifier combinations are somewhat mitigated. The ensemble classifier performs well above the average of the base classifiers for both training and cross-validation AUC. However, the result reaches at best the performance obtained with the best one-class classifier. The product combination rule gives slightly superior results, but the difference is not significant. Considering the additional computational cost induced by using more than one classifier, combinations are not extremely interesting on these datasets if the base classifiers are already good. However, combinations might still be interesting when no knowledge of the methods' performance is known (i.e., with unlabeled data).

The addition of outliers in the training process significantly increased the training AUC of SVDD and GMM I and II on both datasets. However, when looking at the AUC-CV, there is no significant change in the performance of the

Table 3. Comparison of the methods with respect to AUC, AUC-CV, computational cost, robustness and easiness. Easiness is defined as the number of parameters to tune (the smaller the better).

	Heroin dataset		Cocaine dataset	
	best	worst	best	worst
AUC	KNN	GMM I	Gauss	GMM I
AUC-CV	SVDD	GMM I	Gauss	GMM II
Computational cost	Gauss	SVDD	Gauss	SVDD
Robustness	SVDD	KNN	SVDD	GMM II
Easiness	Gauss	GMM I-II	Gauss	GMM I-II

one-class classifiers, given the standard deviations of the results. Oddly enough, the CV performance of SVDD decreases when outliers are presented during training. This, however, is likely a particularity of this specific dataset. The same remarks can be made regarding combination of classifiers. At best, the AUC reaches that of the best classifier.

From these results, it can be inferred that the structure of cocaine data is close to well-separated Gaussian-like clusters. Indeed, the simple Gaussian distribution performed very well, and the prediction performance of all the methods is in general very high. The class separation in heroin data seems to be quite more complicated. All methods, whilst sometimes achieving a good separation (Parzen, KNN, GMM II), have a poor prediction performance. In both cases, the SVDD method has shown to be the most robust. Most importantly, given these results, it is reasonable - at least for cocaine data - to suppose that information regarding the network from which comes a sample might be extracted on the basis of its chemical composition.

6 Conclusion

Several one-class classifiers have been applied and assessed using the AUC criterion for novelty detection in chemical databases of illicit drug seizures. The two datasets have proven very different: far better prediction performance has been obtained with the cocaine dataset, as it could be seen with cross-validation. In most cases, the SVDD method has appeared more robust, even though other methods have outperformed it in some cases. No significant difference was noted between general types of outlier detectors, i.e., density-based, boundary or reconstruction methods. Combinations of classifiers provided better than average results, but at best a similar performance as the best classifier. In addition, providing outliers during training improved the training AUC, but did not change significantly the cross-validation AUC.

In general, results suggest, especially for cocaine, that information regarding the origin of a sample (more precisely, the distribution network) might be extracted from its chemical constituents. This is a very interesting result, since nothing would indicate *a priori* that this is the case. Indeed, products circulating in the same network could come from different producers. Overall, these results have highlighted one-class classification methods that could contribute to the profiling methodology in forensic analysis. Future research topics include considering the time variable. Chemical compositions might exhibit seasonality, and integrating time would likely provide different results.

Acknowledgements

Authors thank S. Ioset for the preparation of the database and B. Petreska for improving the manuscript.

References

1. Esseiva, P., Dujourdy, L., Anglada, F., Taroni, F., Margot, P.: A methodology for illicit heroin seizures comparison in a drug intelligence perspective using large databases. Forensic Science International 132, 139–152 (2003)
2. Esseiva, P., Anglada, F., Dujourdy, L., Taroni, F., Margot, P., Du Pasquier, E., Dawson, M., Roux, C., Doble, P.: Chemical profiling and classification of illicit heroin by principal component analysis. Calculation of inter sample correlation and artificial neural networks, *Talanta* 67, 360–367 (2005)
3. Ratle, F., Terrettaz-Zufferey, A.L., Kanevski, M., Esseiva, P., Ribaux, O.: Pattern analysis in illicit heroin seizures: a novel application of machine learning algorithms. In: Proc. of the 14^{th} European Symposium on Artificial Neural Networks, d-side publi. (2006)
4. Ratle, F., Terrettaz-Zufferey, A.L., Kanevski, M., Esseiva, P., Ribaux, O.: Learning manifolds in forensic data. In: Proc. of the 16^{th} Int. Conf. on Artificial Neural Networks, Springer, Heidelberg (2006)
5. Tong, C., Svetnik, V.: Novelty detection in mass spectral data using a support vector machine method. In: Proc. of Interface 2002 (2002)
6. Markou, M., Singh, S.: Novelty detection: a review - part 1: statistical approaches. Signal Processing 83, 2481–2497 (2003)
7. Markou, M., Singh, S.: Novelty detection: a review - part 2: neural network based approaches. Signal Processing 83, 2499–2521 (2003)
8. Marsland, S.: Novelty detection in learning systems. Neural Computing Surveys 3, 157–195 (2003)
9. Tax, D.M.J.: One-class classification, Ph.D. thesis, University of Amsterdam (2001)
10. Tax, D.M.J., Duin, R.P.W.: Support Vector Data Description. Machine Learning 54, 45–66 (2004)
11. Schölkopf, B., Williamson, R., Smola, A., Shawe-Taylor, J., Platt, J.: Support vector method for novelty detection. Advances in Neural Information Processing Systems 12 (2000)
12. Tax, D.M.J.: Combining one-class classifiers. In: Kittler, J., Roli, F. (eds.) MCS 2001. LNCS, vol. 2096, Springer, Heidelberg (2001)
13. Guéniat, O., Esseiva, P.: Le Profilage de l'Héroïne et de la Cocaïne, Presses polytechniques et universitaires romandes, Lausanne (2005)
14. Tax, D.M.J.: DDtools, the Data Description Toolbox for Matlab, version 1.5.7 (2007)

Variational GTM

Iván Olier and Alfredo Vellido

Department of Computing Languages and Systems (LSI)
Technical University of Catalonia (UPC)
C/. Jordi Girona 1-3, Edifici Omega, Despatx S106
08034 - Barcelona, Spain
{iaolier,avellido}@lsi.upc.edu

Abstract. Generative Topographic Mapping (GTM) is a non-linear latent variable model that provides simultaneous visualization and clustering of high-dimensional data. It was originally formulated as a constrained mixture of distributions, for which the adaptive parameters were determined by Maximum Likelihood (ML), using the Expectation-Maximization (EM) algorithm. In this paper, we define an alternative variational formulation of GTM that provides a full Bayesian treatment to a Gaussian Process (GP)-based variation of GTM. The performance of the proposed Variational GTM is assessed in several experiments with artificial datasets. These experiments highlight the capability of Variational GTM to avoid data overfitting through active regularization.

1 Introduction

Manifold learning models attempt to describe multivariate data in terms of low dimensional representations, often with the goal of allowing the intuitive visualization of high-dimensional data. Generative Topographic Mapping (GTM) [1] is one such model, whose probabilistic setting and functional similarity make it a principled alternative to Self-Organizing Maps (SOM)[2]. In its basic formulation, the GTM is trained within the ML framework using EM, permitting the occurrence of data overfitting unless regularization is included, a major drawback when modelling noisy data. Its probabilistic definition, though, allows the formulation of principled extensions, such as those providing active model regularization to avoid overfitting [3,4].

The regularization methods in [3,4] were based on Bayesian evidence approaches. Alternatively, we could reformulate GTM within a fully Bayesian approach and endow the model with regularization capabilities based on variational techniques [5,6]. In this paper, we define a novel Variational GTM model based on the GTM with GP prior outlined in [3], to which a Bayesian estimation of its parameters is added.

Several preliminary experiments with noisy artificial data were designed to show how Variational GTM limits the negative effect of data overfitting, improving on the performance of the standard regularized GTM [3] and the standard GTM with GP prior, while retaining the data visualization capabilities of the model.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 77–86, 2007.

The remaining of the paper is organized as follows: First, in section 2, an introduction to the original GTM, the GTM with GP prior and a Bayesian approach for the GTM, are provided. This is followed, in section 3, by the description of the proposed Variational GTM. Several experiments for the assessment of the performance of the proposed model are described, and their results presented and discussed, in section 4. The paper wraps up with a brief conclusion section.

2 Generative Topographic Mapping

2.1 The Original GTM

The neural network-inspired GTM is a nonlinear latent variable model of the manifold learning family, with sound foundations in probability theory. It performs simultaneous clustering and visualization of the observed data through a nonlinear and topology-preserving mapping from a visualization latent space in $\Re^L$(with L being usually 1 or 2 for visualization purposes) onto a manifold embedded in the $\Re^D$ space, where the observed data reside. The mapping that generates the manifold is carried out through a *regression function* given by:

$$\mathbf{y} = \mathbf{W}\mathbf{\Phi}\left(\mathbf{u}\right) \tag{1}$$

where $\mathbf{y} \in \Re^D$, $\mathbf{u} \in \Re^L$, $\mathbf{W}$ is the matrix that generates the mapping, and $\mathbf{\Phi}$ is a matrix with the images of S basis functions ϕ_s (defined as radially symmetric Gaussians in the original formulation of the model). To achieve computational tractability, the prior distribution of $\mathbf{u}$ in latent space is constrained to form a uniform discrete grid of K centres, analogous to the layout of the SOM units, in the form:

$$p\left(\mathbf{u}\right) = \frac{1}{K}\sum_{k=1}^{K}\delta\left(\mathbf{u} - \mathbf{u}_k\right) \tag{2}$$

This way defined, the GTM can also be understood as a constrained mixture of Gaussians. A density model in data space is therefore generated for each component k of the mixture, which, assuming that the observed data set $\mathbf{X}$ is constituted by $\mathbf{N}$ independent, identically distributed (i.i.d.) data points $\mathbf{x}_n$, leads to the definition of a complete likelihood in the form:

$$P\left(\mathbf{X}|\mathbf{W},\beta\right) = \left(\frac{\beta}{2\pi}\right)^{ND/2}\prod_{n=1}^{N}\left\{\frac{1}{K}\sum_{k=1}^{K}\exp\left(-\frac{\beta}{2}\left\|\mathbf{x}_n - \mathbf{y}_k\right\|^2\right)\right\} \tag{3}$$

where $\mathbf{y}_k = \mathbf{W}\mathbf{\Phi}\left(\mathbf{u}_k\right)$. From Eq. 3, the adaptive parameters of the model, which are $\mathbf{W}$ and the common inverse variance of the Gaussian components, β, can be optimized by ML using the EM algorithm. Details can be found in [1].

2.2 Gaussian Process Formulation of GTM

The original formulation of GTM described in the previous section has a hard constraint imposed on the mapping from the latent space to the data space due to

the finite number of basis functions used. An alternative approach is introduced in [3], where the regression function using basis functions is replaced by a smooth mapping carried out by a GP prior. This way, the likelihood takes the form:

$$P\left(\mathbf{X}|\mathbf{Z},\mathbf{Y},\beta\right)=\left(\frac{\beta}{2\pi}\right)^{ND/2}\prod_{n=1}^{N}\prod_{k=1}^{K}\left\{\exp\left(-\frac{\beta}{2}\left\|\mathbf{x}_n-\mathbf{y}_k\right\|^2\right)\right\}^{z_{kn}} \tag{4}$$

where: $\mathbf{Z}=\{z_{kn}\}$ are binary membership variables complying with the restriction $\sum_{k=1}^{K} z_{kn}=1$ and $\mathbf{y}_k=(y_{k1},\ldots,y_{kD})^T$ are the column vectors of a matrix $\mathbf{Y}$ and the centroids of spherical Gaussian generators. Note that the spirit of $\mathbf{y}_k$ in this approach is similar to the regression version of GTM (Eq. 1) but with a different formulation: A GP formulation is assumed introducing a prior multivariate Gaussian distribution over $\mathbf{Y}$ defined as:

$$P\left(\mathbf{Y}\right)=\left(2\pi\right)^{-KD/2}\left|\mathbf{C}\right|^{-D/2}\prod_{d=1}^{D}\exp\left(-\frac{1}{2}\mathbf{y}_{(d)}^{T}\mathbf{C}^{-1}\mathbf{y}_{(d)}\right) \tag{5}$$

where $\mathbf{y}_{(d)}$ is each one of the row vectors of the matrix $\mathbf{Y}$ and $\mathbf{C}$ is a matrix where each of its elements is a covariance function that can be defined as

$$\mathbf{C}\left(i,j\right)=\mathbf{C}\left(\mathbf{u}_i,\mathbf{u}_j\right)=\nu\exp\left(-\frac{\left\|\mathbf{u}_i-\mathbf{u}_j\right\|^2}{2\alpha^2}\right),\quad i,j=1\ldots K \tag{6}$$

and where parameter ν is usually set to 1. The α parameter controls the flexibility of the mapping from the latent space to the data space. An extended review of covariance functions can be found in [7]. An alternative GP formulation was introduced in [8], but this approach had the disadvantage of not preserving the topographic ordering in latent space, being therefore inappropiate for data visualization purposes.

Note that Eqs. 3 and 4 are equivalent if a prior multinomial distribution over $\mathbf{Z}$ in the form $P\left(\mathbf{Z}\right)=\prod_{n=1}^{N}\prod_{k=1}^{K}\left(\frac{1}{K}\right)^{z_{kn}}=\frac{1}{K^N}$ is assumed.

Eq. 4 leads to the definition of a log-likelihood and parameters $\mathbf{Y}$ and β of this model can be optimized using the EM algorithm, in a similar way to the parameters $\mathbf{W}$ and β in the regression formulation. Some basic details are provided in [3].

2.3 Bayesian GTM

The specification of a full Bayesian model of GTM can be completed by defining priors over the parameters $\mathbf{Z}$ and β. Since z_{kn} are defined as binary values, a multinomial distribution can be chosen for $\mathbf{Z}$:

$$P\left(\mathbf{Z}\right)=\prod_{n=1}^{N}\prod_{k=1}^{K}p_{kn}^{z_{kn}} \tag{7}$$

where p_{kn} is the parameter of the distribution.

As in [9], a Gamma distribution[1] is chosen to be the prior over β:

$$P(\beta) = \Gamma(\beta|d_\beta, s_\beta) \tag{8}$$

where d_β and s_β are the parameters of the distribution. Therefore, the joint probability $P(\mathbf{X}, \mathbf{Z}, \mathbf{Y}, \beta)$ is given by:

$$P(\mathbf{X}, \mathbf{Z}, \mathbf{Y}, \beta) = P(\mathbf{X}|\mathbf{Z}, \mathbf{Y}, \beta)\, P(\mathbf{Z})\, P(\mathbf{Y})\, P(\beta) \tag{9}$$

This expression can be maximized through evidence methods using the Laplace approximation [10] or, alternatively, using Markov Chain Monte Carlo [11] or variational [5,6] methods.

3 Variational GTM

3.1 Motivation of the Use of Variational Inference

A basic problem in statistical machine learning is the computation of the marginal likelihood $P(\mathbf{X}) = \int P(\mathbf{X}, \Theta)\, d\Theta$, where $\Theta = \{\theta_i\}$ is the set of parameters defining the model. Depending of the complexity of the model, the analytical computation of this integral could be intractable. Variational inference allows approximating the marginal likelihood through Jensen's inequality as follows:

$$\begin{aligned} \ln P(\mathbf{X}) &= \ln \int P(\mathbf{X}, \Theta)\, d\Theta = \ln \int Q(\Theta) \frac{P(\mathbf{X}, \Theta)}{Q(\Theta)} d\Theta \\ &\geq \int Q(\Theta) \ln \frac{P(\mathbf{X}, \Theta)}{Q(\Theta)} d\Theta = F(Q) \end{aligned} \tag{10}$$

The function $F(Q)$ is a lower bound function such that its convergence guarantees the convergence of the marginal likelihood. The goal in variational methods is choosing a suitable form for the density $Q(\Theta)$ in such a way that $F(Q)$ can be readily evaluated and yet which is sufficiently flexible that the bound is reasonably tight. A reasonable approximation for $Q(\Theta)$ is based on the assumption that it factorizes over each one of the parameters as $Q(\Theta) = \prod_i Q_i(\theta_i)$. That assumed, $F(Q)$ can be maximized leading the optimal distributions:

$$Q_i(\theta_i) = \frac{\exp \langle \ln P(\mathbf{X}, \Theta) \rangle_{k \neq i}}{\int \exp \langle \ln P(\mathbf{X}, \Theta) \rangle_{k \neq i}\, d\theta_i} \tag{11}$$

where $\langle\, .\, \rangle_{k \neq i}$ denotes an expectation with respect to the distributions $Q_k(\theta_k)$ for all $k \neq i$.

[1] The Gamma distribution is defined as follows: $\Gamma(\nu|d_\nu, s_\nu) = \frac{s_\nu^{d_\nu} \nu^{d_\nu - 1} \exp^{-s_\nu \nu}}{\Gamma(d_\nu)}$.

3.2 A Bayesian Approach of GTM Based on Variational Inference

In order to apply the variational principles to the Bayesian GTM within the framework described in the previous section, a Q distribution of the form:

$$Q(\mathbf{Z}, \mathbf{Y}, \beta) = Q(\mathbf{Z})\, Q(\mathbf{Y})\, Q(\beta) \tag{12}$$

is assumed, where natural choices of $Q(\mathbf{Z})$, $Q(\mathbf{Y})$ and $Q(\beta)$ are similar distributions to the priors $P(\mathbf{Z})$, $P(\mathbf{Y})$ and $P(\beta)$, respectively. Thus, $Q(\mathbf{Z}) = \prod_{n=1}^{N}\prod_{k=1}^{K}\tilde{p}_{kn}^{z_{kn}}$, $Q(\mathbf{Y}) = \prod_{d=1}^{D}\mathcal{N}\left(\mathbf{y}_{(d)}|\tilde{\mathbf{m}}^{(d)}, \tilde{\mathbf{\Sigma}}\right)$, and $Q(\beta) = \Gamma\left(\beta|\tilde{d}_\beta, \tilde{s}_\beta\right)$. Using these expressions in Eq. 11, the following formulation for the variational parameters $\tilde{\mathbf{\Sigma}}, \tilde{\mathbf{m}}^{(d)}, \tilde{p}_{kn}, \tilde{d}_\beta$ and $\tilde{s}_\beta$ can be obtained:

$$\tilde{\mathbf{\Sigma}} = \left(\langle\beta\rangle \sum_{n=1}^{N}\mathbf{G}_n + \mathbf{C}^{-1}\right)^{-1} \tag{13}$$

$$\tilde{\mathbf{m}}^{(d)} = \langle\beta\rangle\, \tilde{\mathbf{\Sigma}} \sum_{n=1}^{N} x_{nd}\, \langle\mathbf{z}_n\rangle \tag{14}$$

$$\tilde{p}_{kn} = \frac{\exp\left\{-\frac{\langle\beta\rangle}{2}\left\langle\|\mathbf{x}_n - \mathbf{y}_k\|^2\right\rangle\right\}}{\sum_{k'=1}^{K}\exp\left\{-\frac{\langle\beta\rangle}{2}\left\langle\|\mathbf{x}_n - \mathbf{y}_{k'}\|^2\right\rangle\right\}} \tag{15}$$

$$\tilde{d}_\beta = d_\beta + \frac{ND}{2} \tag{16}$$

$$\tilde{s}_\beta = s_\beta + \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\langle z_{kn}\rangle\left\langle\|\mathbf{x}_n - \mathbf{y}_k\|^2\right\rangle \tag{17}$$

where $\mathbf{z}_n$ corresponds to each row vector of $\mathbf{Z}$ and $\mathbf{G}_n$ is a diagonal matrix of size $K \times K$ with elements $\langle\mathbf{z}_n\rangle$. The moments in the previous equations are defined as: $\langle z_{kn}\rangle = \tilde{p}_{kn}$, $\langle\beta\rangle = \frac{\tilde{d}_\beta}{\tilde{s}_\beta}$, and $\left\langle\|\mathbf{x}_n - \mathbf{y}_k\|^2\right\rangle = D\tilde{\mathbf{\Sigma}}_{kk} + \sum_{d=1}^{D}\left(x_{nd} - \tilde{m}^{(kd)}\right)^2$.

Finally, and according to Eq. 10, the lower bound function $F(Q)$ is derived from:

$$F(Q) = \int Q(\mathbf{Z})\, Q(\mathbf{Y})\, Q(\beta) \ln\frac{P(\mathbf{X}|\mathbf{Z}, \mathbf{Y}, \beta)\, P(\mathbf{Z})\, P(\mathbf{Y})\, P(\beta)}{Q(\mathbf{Z})\, Q(\mathbf{Y})\, Q(\beta)} d\mathbf{Z} d\mathbf{Y} d\beta \tag{18}$$

Integrating out, we obtain:

$$\begin{aligned} F(Q) = {} & \langle\ln P(\mathbf{X}|\mathbf{Z}, \mathbf{Y}, \beta)\rangle + \langle\ln P(\mathbf{Z})\rangle + \langle\ln P(\mathbf{Y})\rangle + \langle\ln P(\beta)\rangle \\ & - \langle\ln Q(\mathbf{Z})\rangle - \langle\ln Q(\mathbf{Y})\rangle - \langle\ln Q(\beta)\rangle \end{aligned} \tag{19}$$

where the moments are expressed as:

$$\langle \ln P(\mathbf{X}|\mathbf{Z},\mathbf{Y},\beta)\rangle = \frac{ND}{2}\langle \ln\beta\rangle - \frac{ND}{2}\ln 2\pi - \frac{\langle\beta\rangle}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\langle z_{kn}\rangle\left\langle \|\mathbf{x}_n-\mathbf{y}_k\|^2\right\rangle \quad (20)$$

$$\langle \ln P(\mathbf{Z})\rangle = \sum_{n=1}^{N}\sum_{k=1}^{K}\langle z_{kn}\rangle \ln p_{kn} \quad (21)$$

$$\langle \ln P(\mathbf{Y})\rangle = -\frac{KD}{2}\ln 2\pi - \frac{D}{2}\ln|\mathbf{C}| - \frac{1}{2}\sum_{d=1}^{D}\left\langle \mathbf{y}_{(d)}^T\mathbf{C}^{-1}\mathbf{y}_{(d)}\right\rangle \quad (22)$$

$$\langle \ln P(\beta)\rangle = d_\beta \ln s_\beta - \ln\Gamma(d_\beta) + (d_\beta-1)\langle\ln\beta\rangle - s_\beta\langle\beta\rangle \quad (23)$$

$$\langle \ln Q(\mathbf{Z})\rangle = \sum_{n=1}^{N}\sum_{k=1}^{K}\langle z_{kn}\rangle \ln \tilde{p}_{kn} \quad (24)$$

$$\langle \ln Q(\mathbf{Y})\rangle = -\frac{KD}{2}\ln 2\pi - \frac{D}{2}\ln\left|\tilde{\mathbf{\Sigma}}\right| - \frac{KD}{2} \quad (25)$$

$$\langle \ln Q(\beta)\rangle = \tilde{d}_\beta \ln \tilde{s}_\beta - \ln\Gamma\left(\tilde{d}_\beta\right) + \left(\tilde{d}_\beta-1\right)\langle\ln\beta\rangle - \tilde{s}_\beta\langle\beta\rangle \quad (26)$$

and

$$\langle \ln\beta\rangle = \psi\left(\tilde{d}_\beta\right) - \ln\tilde{s}_\beta \quad (27)$$

$$\left\langle \mathbf{y}_{(d)}^T\mathbf{C}^{-1}\mathbf{y}_{(d)}\right\rangle = \mathrm{tr}\left[\mathbf{C}^{-1}\left(\tilde{\mathbf{\Sigma}} + \tilde{\mathbf{m}}^{(d)}\left(\tilde{\mathbf{m}}^{(d)}\right)^T\right)\right] \quad (28)$$

In the previous expressions, $\Gamma(\cdot)$ are Gamma functions, and $\psi(\cdot)$ is the Digamma function. Details of these calculations can be found in [12].

4 Experiments

4.1 Experimental Design

The main aim of the set of experiments presented and discussed in this section is the preliminary assessment of the robustness of the proposed model in the presence of noise. Moreover, the performance of Variational GTM is compared with that of the standard GTM (with a GP formulation).

The models used in all the experiments were initialized in the same way to allow straightforward comparison. The matrix centroids of the Gaussian generators $\mathbf{Y}$ and the inverse of the variance β were set through PCA-based initialization [1] and the parameters $\{p_{kn}\}$ are fixed and were initialized using the posterior selection probability of the latent node k given data point $\mathbf{x}_n$, defined using Bayes' theorem as:

$$p_{kn} = \frac{\exp\left(-\frac{\beta}{2}\|\mathbf{x}_n-\mathbf{y}_k^*\|^2\right)}{\sum_{k=1}^{K}\exp\left(-\frac{\beta}{2}\|\mathbf{x}_n-\mathbf{y}^*{}_k\|^2\right)} \quad (29)$$

where $\mathbf{y}_k^*$ is the initial value obtained previously for each centroid k. The parameter s_β was set to d_β/β and d_β was initialized to a small value close to 0. For each set of experiments, several values of K and α were used.

4.2 Robustness of the Variational GTM in the Presence of Noise

The goal of this first set of experiments was assessing and comparing the robustness of both the standard GTM using GP and the proposed Variational GTM models in the presence of increasing levels of noise, as well as comparing it to the robustness of the standard regularized GTM with single regularization term [3] trained by EM (GTM-SRT). The artificial data sets used to this end consisted of 700 points sampled from a circumference to which different levels of random Gaussian noise were added (standard deviations of $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35\}$). For each noise level, 10 data sets were randomly generated and used to train every model. All training runs used the following settings: $K = 36$ for all models, $\alpha = 0.1$ for the GTM-GP and the Variational GTM and $d_\beta = 0.01$ for the Variational GTM. Furthermore, the number of basis functions for GTM-SRT was set to 25. Different values of K and α were considered with similar results.

Two measures were employed to gauge the regularization capabilities of the models: The mean square error between the centroids $\{\mathbf{y}_k\}$ and the underlying circumference without noise, and the standard deviation of the square error. The results for these measures, displayed in Fig. 1, indicate that, as the levels of noise increase, the mean and standard deviation square errors grow to be much higher for the standard GTM using GP than for the proposed Variational GTM, although in the case of the mean error this difference cannot be clearly appreciated for very low levels of noise. Furthermore, Variational GTM is shown to outperform GTM-SRT at all noise levels, while being far less sensitive to the increase of such levels.

These results are a preliminary but clear indication that the proposed Variational GTM provides better regularization performance than both the standard GTM using GP and GTM-SRT. This is neatly illustrated in Fig. 2, for the first two models, where two samples of the artificial data sets used in this experiments and their corresponding results (represented by the connected centroids) are displayed. Although at low noise levels, both models perform similarly, at higher levels the standard GTM using GP fits the noise to a great extent, whereas Variational GTM is much less affected by it and is capable of reproducing the underlying data generator far more faithfully. This should lead to a model with better generalization capabilities.

4.3 Data Visualization Using Variational GTM

A second set of experiments was carried out with the aim of verifying the topographic preservation capabilities of the proposed Variational GTM and consequently, its data visualization capabilities on a low-dimensional discrete latent space. For that, an artificial data set consisting of 12 hetereogenously separated

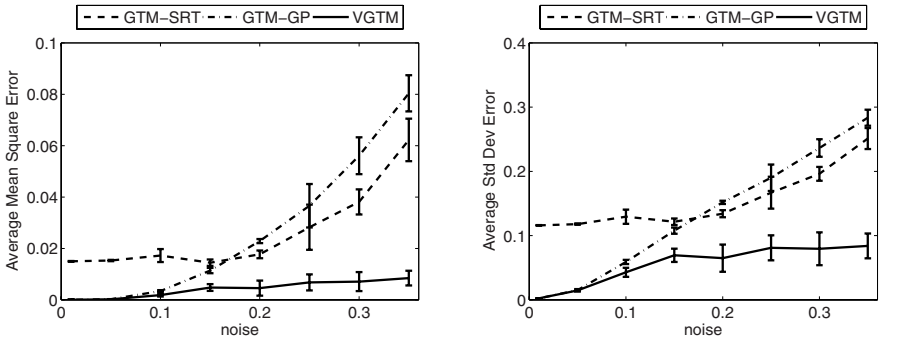


Fig. 1. Plots of the average mean square error between the centroids $\{\mathbf{y}_k\}$ and the theorical circumference whithout noise (left plot) and the average standard deviation of the square error (right plot) for GTM-SRT (dashed line), for the standard GTM using GP (dashed-dotted) and the proposed Variational GTM (solid). The vertical bars indicate the standard deviation of these averages.

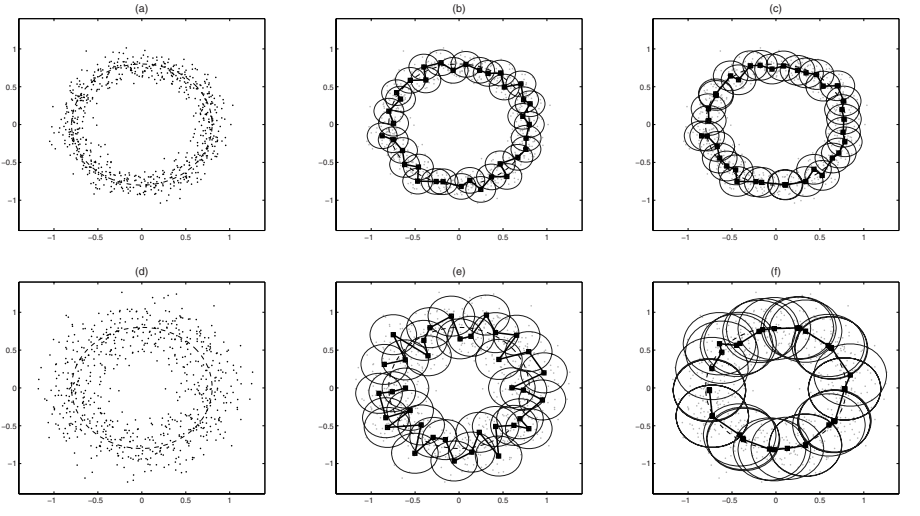


Fig. 2. Left column: two of the artificial data sets generated from a circumference (dashed line) to which noise of levels 0.1 (top row) and 0.25 (bottom row) was added. Middle column: including results after training using standard GTM with GP prior. Right column: including results after training using the proposed Variational GTM. The resulting manifold embedded in the data space is represented by the connected centroids $\{\mathbf{y}_k\}$ (filled squares) in the centres of circles of radius $2\sqrt{\beta^{-1}}$) (common standard deviation).

clusters was generated by means of an equivalent number of radial Gaussian distributions. The following settings were used to train the model: $K = 64$, $\alpha = 0.1$ and $d_\beta = 0.01$. The resulting data visualization is accomplished through the

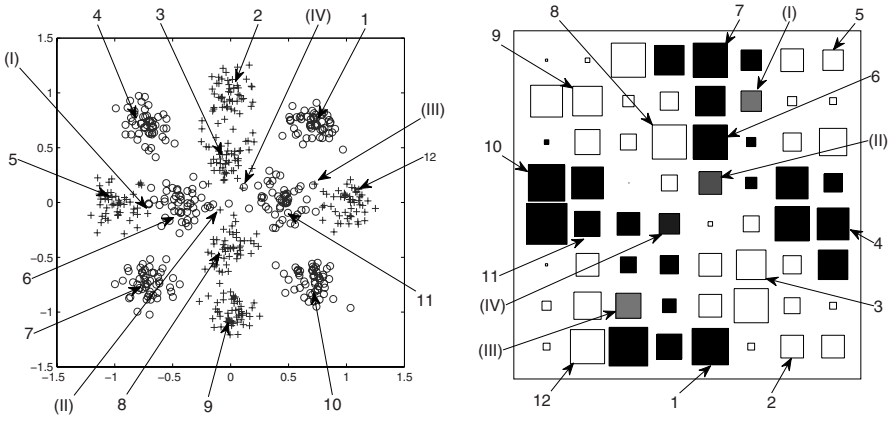


Fig. 3. (Left) 600 points randomly sampled for 12 Gaussians, forming clusters artificially labeled as 'o' and '+'. These labels are only used for visualization purposes and were not included in the training. (Right) The resulting membership map corresponding to the mode projection into a latent space of $K = 64$ latent points (represented as squares). The relative size of each square is proportional to the number of data assigned to its respective latent point and the hue of gray indicates the ratio of the data belonging to each cluster label: white for '+' and black for 'o'. The arabic numbers (1 to 12), signaling specific data points, show how their neighbouring relations are preserved in the latent space. The romanic numbers (I to IV) signal with ambiguous cluster allocation. They are all mapped into *grey points* in latent space.

membership map generated by means of the mode projection [1] of the data into the latent space, given by $\mathbf{u}_n^{\texttt{mode}} = \underset{k}{\texttt{argmax}}\,(\tilde{p}_{kn})$, where the variational parameter $\tilde{p}_{kn}$ was used.

The data set and its corresponding membership map are displayed in Fig. 3, where several interesting data points, some of these placed well within the clusters and others in the edge between two clusters, are singled out for illustration. It is clear that their representation in latent space faithfully preserves the existing topographic ordering and neighbouring relations in data space.

5 Conclusions

Details of a variational formulation of GTM have been provided in this paper. Through several experiments, Variational GTM has been shown to endow the model with effective regularization properties, enabling it to avoid, at least partially, fitting the noise and, therefore, enhancing its generalization capabilities. This regularization has been shown to be more effective than that provided by the standard GTM with GP formulation and the standard regularized GTM.

The experiments reported in this brief paper are necessarily limited by space availability and therefore preliminary. A much more detailed experimental design, including more datasets spanning a wider range of characteristics, as well

an explicit testing of its generalization capabilities, would be required to complete the assessment of the model. The current study should be understood as a first step towards that end.

A variational treatment of parameter α is difficult and, therefore, it was fixed a priori in the reported experiments. However, an interesting approach to its calculation in the context of variational GP classifiers, using lower and upper bound funtions, was presented in [13] and could be considered in future work with the proposed Variational GTM.

References

1. Bishop, C.M., Svensén, M., Williams, C.K.I.: GTM: The Generative Topographic Mapping. Neural Comput. 10(1), 215–234 (1998)
2. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Berlin (2001)
3. Bishop, C.M., Svensén, M., Williams, C.K.I.: Developments of the Generative Topographic Mapping. Neurocomputing 21(1–3), 203–224 (1998)
4. Vellido, A., El-Deredy, W., Lisboa, P.J.G.: Selective smoothing of the Generative Topographic Mapping. IEEE T. Neural Networ. 14(4), 847–852 (2003)
5. Beal, M.: Variational algorithms for approximate Bayesian inference. PhD thesis, The Gatsby Computational Neuroscience Unit, Univ. College London (2003)
6. Jakkola, T., Jordan, M.I.: Bayesian parameter estimation via variational methods. Stat. Comput. 10, 25–33 (2000)
7. Abrahamsen, P.: A review of Gaussian random fields and correlation functions. Technical Report 917, Norwegian Computing Center, Oslo, Norway (1997)
8. Utsugi, A.: Bayesian sampling and ensemble learning in Generative Topographic Mapping. Neural Process. Lett. 12, 277–290 (2000)
9. Bishop, C.M.: Variational principal components. In: Proceedings Ninth Intern. Conf. on Artificial Neural Networks, vol. 1, pp. 509–514 (1999)
10. MacKay, D.J.C.: A practical Bayesian framework for back-propagation networks. Neural Comput. 4(3), 448–472 (1992)
11. Andrieu, C., de Freitas, N., Doucet, A., Jordan, M.I.: An introduction to MCMC for machine learning. Mach. Learn. 50, 5–43 (2003)
12. Olier, I., Vellido, A.: A variational Bayesian formulation for GTM: Theoretical foundations. Technical report, Technical University of Catalonia (UPC) (2007)
13. Gibbs, M., MacKay, D.J.C.: Variational Gaussian process classifiers. IEEE T. Neural Networ. 11(6), 1458–1464 (2000)

Skill Combination for Reinforcement Learning

Zhihui Luo, David Bell, and Barry McCollum

School of Electronics, Electrical Engineering and Computer Science
Queen's University Belfast
{zluo02, da.bell, b.mccollum }@qub.ac.uk

Abstract. Recently researchers have introduced methods to develop reusable knowledge in reinforcement learning (RL). In this paper, we define simple principles to combine skills in reinforcement learning. We present a skill combination method that uses trained skills to solve different tasks in a RL domain. Through this combination method, composite skills can be used to express tasks at a high level and they can also be re-used with different tasks in the context of the same problem domains. The method generates an abstract task representation based upon normal reinforcement learning which decreases the information coupling of states thus improving an agent's learning. The experimental results demonstrate that the skills combination method can effectively reduce the learning space, and so accelerate the learning speed of the RL agent. We also show in the examples that different tasks can be solved by combining simple reusable skills.

1 Introduction

Reinforcement learning offers a fundamental framework for intelligent agents to improve their behavior through interacting with the environment. In a delayed feedback environment, RL proves to be a feasible way to train an agent to perform at a high standard. Recently within the RL domain, there has been increased interest in transferable learning which attempts to reuse learned control knowledge across different problems. In recent research, one outlined method is to build portable knowledge in the option framework using agent space [1]. Another method uses rule transfer to assist learning that target tasks from source tasks [2]. These methods show promising results. However, they do not provide methods to represent tasks on a higher level in reinforcement learning. In this paper, we address the problem of how to represent and combine learned skills to make it suitable for different tasks. Thus an agent can learn quickly. Moreover, this assists the agent to construct a suitable and more compact representation for tasks, which guides the agent's learning process.

2 Skills in Reinforcement Learning

In this section we introduce the problem domain. In the context of reinforcement learning, a problem domain is an environment which consists of more than one

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 87–96, 2007.

sub-problem, and each of the sub-problems can be learned by an agent independently. An agent attempts to solve a sub-problem as a task in the problem domain.

Given an environment modeled as MDP - Markov Decision Process [3], a problem domain can be denoted as $M = (S, A, P, R)$. State space S represents a finite set of states in the environment. A set of actions that can be executed by the agent is denoted as A. R is denoted as the reinforcement values receive from environment, and P is representing the state transition probability.

Environmental features are properties that can be observed and measured by the agent. In this paper, we concentrate on those environment features which can be independently perceived and interacted with by the learning agent. Hence, the agent can learn to interact with each of the features separately to form skills. Suppose there are k numbers of features in the environment. Each of these features indicates an object in the environment. When the agent perceives the ith object, the agent naturally forms a state space representing the condition of this specific object in the environment. The state space of the object is denoted as S^i. All these k number of objects form an MDP's state space S, so we have:

$$S = \left\{ S^1, S^2, S^3, \cdots S^k \right\} = \left\{ S^i \mid i = 1, ..., k \right\}$$

From this definition, we know S^i is a sub-state space of S, so $S^i \subset S$. Each of these sub-state spaces represents an independent feature of the environment.

If an agent can learn each S^i's value separately without being affected by or interfering with other sub-states spaces, we call this learning process skill training. Formally, a skill is an ability of the agent, learned or acquired through training, to perform actions to achieve a desired goal within a problem domain. So the agent can be trained to develop a skill to execute optimal actions in a sub-state space S^i.

$$Skill^i : S^i \mapsto A^i$$

A skill of an agent can be provided by a supervised instructor or learned by unsupervised learning [4]. In this paper, our experimentations are in the reinforcement learning context, so all skills are trained using RL algorithms. In the next section we present our methods to use the combination of these skills to solve more complex tasks within a problem domain.

3 Skills Combination

In this section, we will define three basic principles to combine skills. These principles provide fundamental operators between two skills on a higher level of representation, which are independent from low level skill learning.

Sequential Combination: If skill g finishes then consequently skill h begins. We denote the sequential combination as:

$$C_{seq} = g \rightarrow h \tag{1}$$

The sequential operator $\rightarrow$ is weakly constrained between two skills. The requirements for it are the following properties:

1. Termination exists - if β is the termination condition of skill g and S^g is the state space of skill g, then there at least exists one terminal state s^g satisfying condition β: $\exists s^g \subset S^g, where\ \beta\ is\ satisfied$

2. Initialization exists - similarly, if ω is the initialization condition of skill h, and S^h is the state space of skill h, then there at least exists one initialization state s^h satisfying ω: $\exists s^h \subset S^h, where\ \omega\ is\ satisfied$

3. State inheritable - if terminal state s^g is a sub-state of problem domain's state s, then initialization state s^h is also a sub-state of s. This property ensures termination state s^g can be inherited by initialization state s^h of the consequences skill: $(s^g \subset s) \Rightarrow (s^h \subset s)$

4. A sequential combination links one skill after another. When skill g finishes, the condition of the agent will pass to subsequence skill h. Consequently the agent's action is determined by the subsequence skill, the features relating to skill g can be ignored.

Concurrent Combination: Skill g and skill h will be performed by the agent concurrently. We denote concurrent combination as:

$$C_{con} = g \cap h \tag{2}$$

The concurrent operator $\cap$ is a tight binding between the skills, and satisfies the following properties:

1. Combined action – if skill g generates action a and skill h generates action b at the same time, then the combined action is denote as: $\overline{ab}$

2. State compatible- if s' is the successor state after a combined action $\overline{ab}$, then the successor state s' must be a state in both state spaces S^h and S^g. This property makes sure that when two skills are executed at the same time by the agent, they won't generate a new state that does not exist in both of the skills. $\forall s' \subset (S^h, S^g)$

3. When skills are concurrently combined, the agent's action is determined by the combined action $\overline{ab}$.

Optional Combination: Nondeterministic branch - skill g or skill h to be selected. We denote optional combination as:

$$C_{opt} = g \cup h \tag{3}$$

The optional operator $\cup$ is a weak binding of skills. It has the following properties:

1. Initialization exists - if β, ω are the initialization conditions of skill g and h, and S^g, S^h are the state spaces of skill g and h, then at least there exists one initialization state s satisfies β and ω at the same time: $\exists s \subset S^g, where\ \beta\ is\ satisfied$ and $\exists s \subset S^h, where\ \omega\ is\ satisfied$

2. Comparable selection – if β, ω are the initialization conditions of skills g and h, then β and ω can be compared: $\beta \geq \alpha$ or $\alpha \geq \beta$

3. After comparing the initialization conditions of skills, the agent will select and perform one of the skills. Hence the agent's action is determined by the selected skill. The features relating to the other skills can be ignored.

A common example of using skill combination is in the car driving domain. Suppose a driver learns the following skills: turnRight, turnLeft, goFoward, slowDown and signalLight. Then the driver can combine these skills to solve different tasks. For example, before a crossroads, a car driver needs to determine which way to go. This task can be represents as:

$$\text{slowDown} \cap (\text{turnRight} \cup \text{turnLeft} \cup \text{goFoward})$$

If the driver decides to go right, the composite task would be:

$$\text{signalLight} \rightarrow \text{turnRight}$$

If all the driving skills are well trained, the driver can use them to solve complex tasks on the road. We can see from the examples that by using skill combination, a more flexible knowledge representation can be used to represent different tasks in standard reinforcement learning method, and skills can be reused on different tasks.

4 Experiments

4.1 Cat and Mouse Domain Introduction

In the cat and mouse domain, we have a reinforcement learning agent - Mouse (A). We also have predator - Cat (C), food - Cheese (Z) and Home (H). The cat and mouse can each move one step forward in any direction – vertically, horizontally or diagonally. They can also choose to stay still. So there are nine possible movements. From the point of view of the mouse agent, objects in its environment - cat, cheese and home - are features of the domain.

In order to learn skills, the agent has sensors to detect the direction and distance from every feature in the environment. In this experimental setting, the agent has sensors to perceive the cat, cheese, and home in the environment. The agent can detect an object in 8 directions in the grid world (See Fig. 1). The object gives signal sharing its distance to the agent. Using the direction and distance data from the sensor, the agent can form the state space of a basic skill related to that feature.

4.2 Skill Training

Skills are independent, low-level learning problems in the environment, so they can be acquired before being applied to tasks. We design three types of skills: skill to get cheese, skill to keep away from cat and skill to store cheese. In the experiments, all skills are trained in a 10*10 grid environment using the Q-learning algorithm [5, 6] with the following standard learning parameters: ε-greedy policy, alpha (α) =0.1, gamma (γ) =0.9, epsilon (ε) =0.2. To facilitate our subsequence experiments with more comparative results, all these skills are trained to different degrees. Each of the

skills is saved on the following training episodes: 200, 400, 600, and 800. So in our task learning experiments below, we can equip the agent with skills that have different levels of proficiency. All experiences of trained skills will be saved for further use on the subsequence tasks.

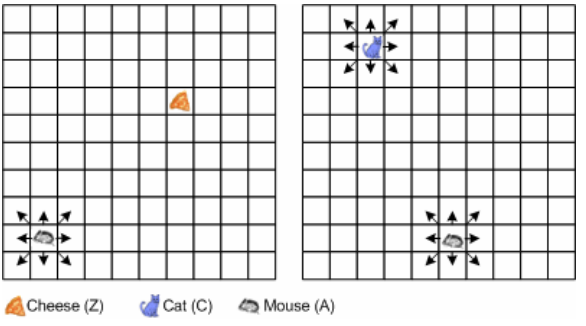


Fig. 1. Left: a skill training environment for getting cheese. Right: a skill training environment for keeping away from cat.

4.3 Task 1: Get Cheese

In this section, we demonstrate how to use skills in a task. In task 1 (See figure 2 left), the mouse agent target is to complete a 2-skill task to get a piece of cheese in a 20*20 grid environment, in the mean time the agent should try its best to avoid being caught by an approaching cat. To finish this task, the agent needs to combine the concurrent skills of getting the cheese and keeping away from the cat. Following the definition of skill's concurrent combination above (2), this task can be expressed as:

$$GC(Z) \cap KA(C)$$

$GC(Z)$ represents a skill of getting the cheese and $KA(C)$ represents a skill of keeping away from the cat. They are concurrently performed by the agent, so the concurrent operator $\cap$ is used to express the relationship between the two skills.

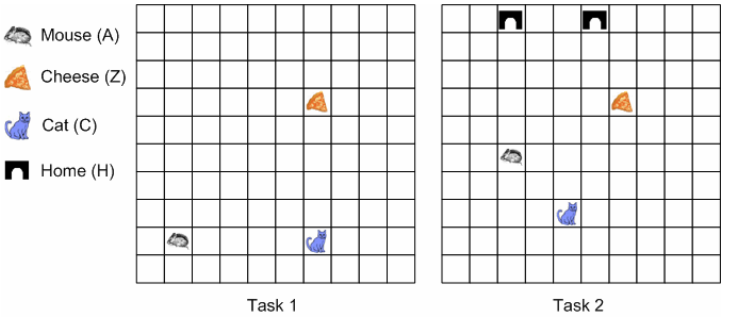


Fig. 2. Left: An example environment for task 1. Right: An example environment for task 2.

Experimental Settings

Experiments of task 1 are carried out on randomly generated grid environments. The initial positions of the mouse, cat and cheese are randomly selected for each experiment. The mouse agent will receive a reward of +100 when it gets a cheese. If the cat catches the mouse, a punishment of -100 will be received by the agent, and the episode will end. Any other action taken by the mouse will get a step penalty of -1.

We first evaluate the performances of three different types of learning agents. The first type is the agents without skills. Agents without skills will perform Q-learning on the task and basically they seek to learn task 1 in its entirety. The second type is agents with skills, but their skills are not trained before performing the task. The third type is agents with trained skills. For each type of agent, we generate 20 randomly initialized experiments. The experiments are stopped at episode 30000 and the data is recorded. Figure 3 (left) illustrates the learning curve of the experiments.

To compare agents with different experiences of skills, our second experiment of task 1 involved equipping the agent with different proficiencies of trained skills. We ran the agent with no trained skills and then with trained skills of 200, 400, 600 and 800 episodes. Figure 3 (right) compares the learning curves of the experiments.

Experimental results

In this part, we show and explain the experiment results of task 1.

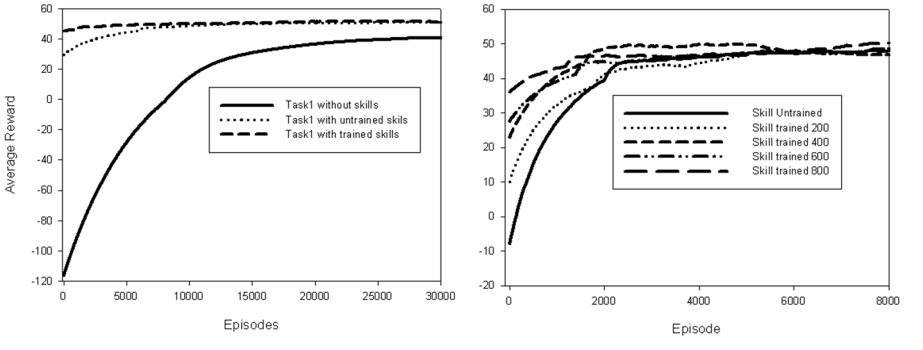


Fig. 3. Left: task 1 performance comparison of agents with combined skill, without skills and with trained skills. Right: task 1 performance comparison of agents with no trained skills and with trained skills of 200, 400, 600 and 800 episodes. All curves are smoothed using sampling proportion 0.2.

Figure 3 (left) compares the average learning curves of agents with combined skills, without skills and with trained skills. We can see from the left figure that agents with sufficient trained skills show significant better performance in task 1 than the other type of agents, especially at the early stages of learning (episodes 1- 5000). Agents using skills with no prior experience also show much better performance than agents without skills. Although these skills are not trained before applying to task 1, our skill combination method gives the agents a higher level of expression of the task. In a long run over 20000 episodes, agents with skills also gained more reward on average than standard learning agents. Figure 3 (right) compares average learning

curves of agents with different experiences of skills. This graph shows that gradually agents with more experience on skills perform better at the early state of learning. These experiments show that the skill combination method is stable for learning.

4.4 Task 2: Cheese Storage

Task 2 is more complex than task 1 in the cat and mouse domain (See figure 2 right). In this task, the mouse agent is required to get the cheese in the grid environment and then carry the cheese to the mouse's home. In the setting of task 2, an additional two homes - home1 (H_1) and home2 (H_2) are added into the environment. When a mouse is carrying a cheese, either of the two homes can be selected to store the cheese.

From this task description, we can define task 2 as follow:

$$GC(Z) \cap KA(C) \tag{4}$$

$$PC(H_1) \cap KA(C) \tag{5}$$

$$PC(H_2) \cap KA(C) \tag{6}$$

$$PC(H_1) \cup PC(H_2) \tag{7}$$

$$GC(Z) \rightarrow PC(H) \tag{8}$$

Expression (4) actually is the same as task 1. Expressions (5) and (6) represent concurrent skill combination of storing a cheese and keeping away from cat. Expression (7) represents the optional combination of storing cheese into home 1 or home 2. Expression (8) represents the sequential combination of getting the cheese and storing the cheese into a home. Hence, we can use knowledge reasoning methods to get the following expression of task 2:

$$(GC(Z) \rightarrow (PC(H_1) \cup PC(H_2))) \cap KA(C)$$

Experimental Settings

We carried out experiments of task 2 on randomly generated grid worlds. The initial positions of mouse, cat, cheese and homes are randomly selected during each of the experiments. The mouse agent receives a medium reward of +10 when it gets a cheese, and when it successfully puts the cheese into one of the homes, it gets a positive reward of +100. If the cat catches the mouse, a punishment of -100 will be received by the agent, and the episode ends. Any other actions taken by the mouse gets a step penalty of -1.

As task 1, we did two types of experiments in task 2. The first type of experiment compares agents with skills, without skill and with trained skills. Figure 4 (left) shows the learning curve of the first type of experiments. The second type of experiment compares agents with skills that have different levels of proficiency. Figure 4 (right) shows the graph of results.

Experimental Results

In this part, we present and explain the experimental results of task 2.

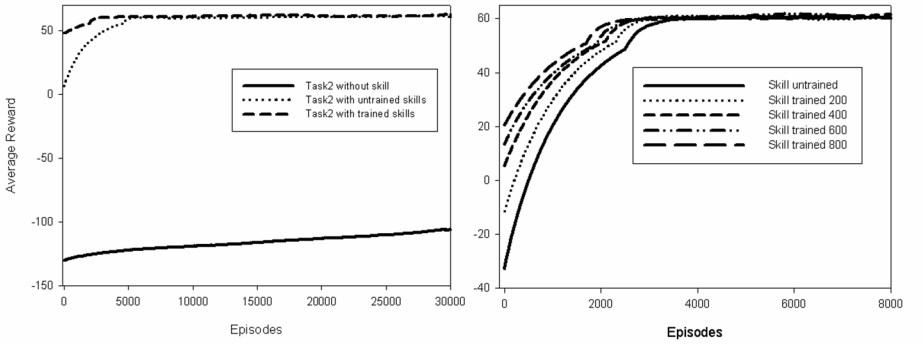


Fig. 4. Left: Task 2 performance comparison of agents with skills, without skill and with trained skills. Right: Task 2 performance comparison of agents with no trained skills and with trained skills of 200, 400, 600 and 800 episodes. All curves are smoothed using sampling proportion 0.2.

Figure 4 (left) compares average learning curves of agents with combined skills, without skills and with trained skills. In task 2, we find that agents using skills, no matter if they are trained or not, show remarkably better performance than agents without skills throughout the whole test (from episode 1-30000). Agents using skills with no prior experience reach a high standard of average reward (roughly +58) before episode 5000, while agents without skills still have a large amount of negative reward (roughly -122) at that stage. Even in a long run to episode 30000, agents without skills only make a small improvement to reward -106 which is far from optimal. Figure 4 (right) compares average learning curves of agents in task 2 with different skill experience. This graph shows that gradually agents with more experience on skills perform better at the early stage of learning. These experiments show that the skill combination method exhibits stable improvement for learning.

4.5 Compare Task 1 and Task 2

We will compare the learning efficiency of task 1 and task 2 in this section.

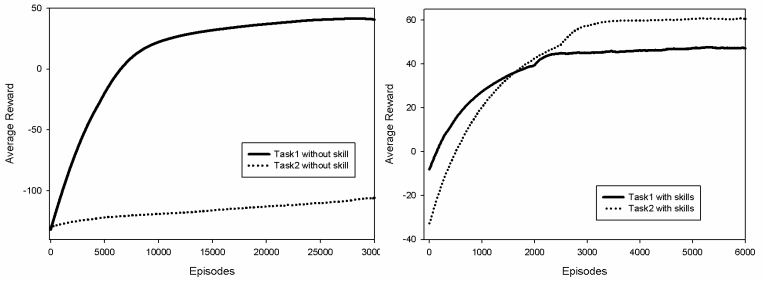


Fig. 5. Left: Compare learning curve of agents without skill between task1 and task2. Right: Compare learning curve of agents with skill between task1 and task2.

Figure 5 (left) compares the learning curves of agents without skills between task 1 and task 2. The graph shows that task 2 is a much more difficult problem compared to task 1. Although only 2 more features (Home 1 and Home 2) were added into task 2, the performance of the agent decreases dramatically. Agents without skills only improve nearly +1 more average reward in every 5000 episode of learning. On the other hand, task 2 is not a hard problem to agents with skills. Figure 5 (right) show that skilled agent performs very well in task 2. They reach a high standard of average reward between episodes 4000 to 5000. The learning curve of task 2 shows a steeper rise than task 1 and also gives a higher average reward when the learning is stable. The reason is that the settings of task 2 sets a medium reward of +10 when the agent gets cheese and +100 for the agent putting cheese to the home, so the sum of task 2's reward is +10 more than task 1's.

5 Discussion

From the previous section's comparison, we find that agents with skills show much better performance than agents without skills, especially when the problem becomes complex as in task 2. We use the big *O* notation to describe the asymptotic bound of the agent's state space and use *n* to denote the number of learning features in the task and *k* as the size of a learning feature. Then the scalability of state space of agents with skills can be denoted as $O(kn)$ which is a linear function of *n* and *k*. Agents use appropriate skills to deal with each of the learning features in the problem. The relations of these learning features are abstracted to a higher level which can be solved by combined skills. So the size rise of state space from task 1 to task 2 is a *linear growth*. On the other hand, the state space of agents without skill is $O(k^n)$. Every new learning feature added to the problem will make the state space *grow exponentially* in respect of the feature's size *k*. We know that $O(kn) \ll O(k^n)$ when $k>10$, $n>2$. So the learning efficiency of agent with skills is much better than agents without skills.

The trained skills in our work are reusable in the same problem domain. Trained skills are given to agents in task 1 and task 2 in the cat and mouse domain, and they generate better performance than an agent with no skill. Skills represent transferable knowledge to different tasks in the problem domain. In recent work, [7] proposes a method to construct mapping to value function based transfer in RL. Using policy based transfer, [8] utilizes transfer through inter-task mappings to construct a transfer functional from a source task to a target task. Another approach is to use shaping. [9] introduces the use of learned shaping reward in RL tasks, where an agent uses prior experience on a sequence of task to learn a portable predictor. These researches show that using transfer learning can markedly reduce learning time on different task.

Our work can also be viewed as a kind of model based learning method which uses tasks to represent problems at a high level and uses skills to construct a model of tasks. [6] provides an introduction of models and planning. A recent approach described in [10] uses a designed environment model to generate a selection of policy reinforcement learning.

Finally, we define principles for combining skills in RL, but we did not specify how these principles are implemented in different problem domain. We need a flexible representation at a high level, so that implementation can be carried out on further problems.

6 Conclusion and Future Work

In this paper, we present a skills combination method for high level knowledge representation in reinforcement learning. The experimental results and analysis shows that our method gives the learning agents a flexible way to express and solve different tasks. Our method can also reduce the learning space of the problems which significantly improves the learning speed of agent. In this study, agents use skills as predefined knowledge. But we believe that agent can automatically generate task representation. This is a topic for our future work.

References

1. Konidaris, G.D., Barto, A.G.: Building Portable Options: Skill Transfer in Reinforcement Learning. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence 2007, Hyderabad, India, January 6-12, 2007 (2007)
2. Taylor, M.E., Stone, P.: Cross-Domain Transfer for Reinforcement Learning. In: ICML 2007. Proceedings of the Twenty-Fourth International Conference on Machine Learning (2007)
3. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley-Interscience, Chichester (2005)
4. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall Series in Artificial Intelligence, pp. 763–788 (2003)
5. Watkins, C., Dayan, P.: Q-Learning. Machine Learning 8(3-4), 279–292 (1992)
6. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
7. Liu, Y., Stone, P.: Stone. Value-Function-Based Transfer for Reinforcement Learning Using Structure Mapping. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence (2006)
8. Taylor, M.E., Whiteson, S., Stone, a.P.: Transfer via InterTask Mappings in Policy Search Reinforcement Learning. In: The Sixth International Joint Conference on Autonomous Agents and Multiagent Systems, May 2007 (2007)
9. Konidaris, G., Barto, A.: Autonomous Shaping: Knowledge Transfer in Reinforcement Learning. In: Proceedings of the Twenty Third International Conference on Machine Learning, Pittsburgh (2006)
10. Kalyanakrishnan, S., Stone, P., Liu, Y.: Model-based Reinforcement Learning in a Complex Domain. In: RoboCup-2007: Robot Soccer World Cup XI, Springer, Berlin (2007)

A New Recurring Multistage Evolutionary Algorithm for Solving Problems Efficiently

Md. Monirul Islam[1,2], Mohammad Shafiul Alam[3], and Kazuyuki Murase[2]

[1] Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh
[2] Department of Human and Artificial Intelligence Systems, Graduate School of Engineering, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan
[3] Department of Computer Science and Engineering, Ahasanullah Univeristy of Science and Technology, Bangladesh

Abstract. This paper introduces a new approach, called recurring multistage evolutionary algorithm (RMEA), to balance the explorative and exploitative features of the conventional evolutionary algorithm. Unlike most previous work, the basis of RMEA is repeated and alternated executions of two different stages i.e. exploration and exploitation during evolution. RMEA uses dissimilar information across the population and similar information within population neighbourhood in mutation operation for achieving global exploration and local exploitation, respectively. It is applied on two unimodal, two multimodal, one rotated multimodal and one composition functions. The experimental results indicated the effectiveness of using different object-oriented stages and their repeated alternation during evolution. The comparison of RMEA with other algorithms showed its superiority on complex problems.

Keywords: Evolutionary algorithm, exploration, exploitation and optimization problem.

1 Introduction

Evolutionary algorithms, such as evolution strategies (ES) [1], evolutionary programming (EP) [3,4], and genetic algorithms (GAs) [6], have been widely used in global optimization problems that have a great importance in science, engineering and business fields. The basic difference between EP (or ES) and GAs is the evolutionary operator used in producing offspring. Although EP was first introduced as an approach to artificial intelligence, it was extended later and applied successfully to many practical and continuous parameter optimization problems.

The mutation is the main operator in EP [3,4]. Thus a number of innovative mutation operators e.g. Cauchy mutation [14], a combination of Cauchy and Gaussian mutation [2] and Lévy mutation [8] have been proposed to improve the performance of EP. The aim of such mutations is to introduce large variations in evolving individuals so that they can explore a wider region of the search space globally. This means that the improvement has been sought by increasing the

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 97–106, 2007.

exploration capability of EP, which is very much important for problems with many local optima. However, both global exploration and local exploitation are necessary during evolution depending on whether an evolutionary process gets stuck into local optima or finds some promising regions in the search space. It is, therefore, necessary for an algorithm to maintain a proper balance between exploration and exploitation in finding a good near-optima for complex problems.

This paper introduces a new recurring multistage evolutionary algorithm (RMEA) based on mutation. RMEA attempts to maintain a proper balance between exploration and exploitation by its two recurring stages. It uses object-oriented mutation operators and selection strategies in achieving exploration-exploitation objectives of the two stages. Although there are many algorithms that use GAs [6] for exploration and local methods for exploitation (see review paper [7]), RMEA is the first algorithm, to our best knowledge, that uses only mutation. Its emphasis on using different recurring stages can increase the solution quality of an evolutionary algorithm.

RMEA differs from most EP [4] based algorithms (e.g. [2], [8] and [14]), and memetic algorithms [11] on two aspects. First, RMEA emphasizes on achieving global exploration and local exploitation by executing object-oriented operations repeatedly and alternatively. This approach is different from the one used in EP [4] and memetic algorithms [11]. EP does not execute exploration and exploitation operations separately rather it uses single stage execution module with self-adaptation rules. Memetic algorithms generally executes exploration and exploitation operations one by one or they use some heuristics to decide when and where exploration and exploitation operations are to be executed within the evolutionary process. The difficulty of this approach lies in avoiding deep local optima and realizing the potentials of very promising regions or finding good heuristics.

Second, RMEA uses only mutation operators for achieving the exploration and exploitation objectives of an evolutionary process. It is argued in this paper that mutation is a better candidate than crossover for achieving exploration and exploitation objectives. For example, the global exploration and local exploitation can easily be achieved by using mutations with large and small step sizes, respectively. Memetic algorithms generally use GAs [6] for exploration and local search methods or specially designed crossover for exploitation. However, it would be difficult to find a proper integration method in order to achieve synergistic effect of different methods or operators.

The rest of this paper is organized as follows. Section 2 describes RMEA in detail and gives motivations and ideas behind various design choices. Section 3 presents experimental results of RMEA and their comparison with other work. Finally, section 4 concludes with a summary of the paper and a few remarks.

2 Recurring Multistage Evolutionary Algorithm

A recurring two-stage evolutionary approach based on mutation is adopted in RMEA to ensure a proper balance between global exploration and local exploitation during evolution. The exploration and exploitation stages in RMEA use the

distance of dissimilar and similar individuals, respectively, as the standard deviation for mutation. In fact, the motivation behind such an idea is inspired by observing the following important facts.

- Exploration is a non-local operation so mutation involving the distance of dissimilar individuals, which is expected to be large, may guide an evolutionary process toward the unexplored regions of a search space.
- Exploitation is a local operation so mutation involving the distance of similar individuals, which is expected to be small, may guide the evolutionary process toward the local neighborhoods.

The major steps of RMEA can be described as follows.

Step 1) Generate an initial population consisting of μ individuals. Each individual $\vec{x_i}$, $i = 1, 2, \ldots, \mu$, is represented as a real valued vector. It is consisted of n independent components

$$\vec{x_i} = \{x_i(1), x_i(2), \ldots x_i(n)\} \tag{1}$$

Step 2) Randomly initialize two parameters K_1 and K_2 within a certain range. These user specified parameters control the behavior of RMEA by defining how many generations the exploration and exploitation operations to be executed repeatedly during evolution.

Step 3) Calculate the fitness value of each individual $\vec{x_i}$, $i = 1, 2, \ldots, \mu$, in the population based on the objective function. If the best fitness value is acceptable or the maximum number of generations has been elapsed, stop the evolutionary process. Otherwise, continue.

Step 4) Repeat the following steps 5-8 for K_1 generations, which constitutes a single pass of the exploration stage.

Step 5) For each individual $\vec{x_i}$, $i = 1, 2, \ldots, \mu$, select ϕ individuals from the population in such a way that the fitness value of them is very different comparing to that of $\vec{x_i}$. The ϕ individuals, therefore, can be considered as *strangers* i.e., distant individuals for $\vec{x_i}$. Here the parameter ϕ is specified by the user.

Step 6) Create μ offspring by applying mutation on each individual $\vec{x_i}$, $i = 1, 2, \ldots, \mu$, of the population. Each individual $\vec{x_i}$ creates a single offspring $\vec{x_i}'$ by: for $j = 1$ to n

$$k = 1 + r_j \; mod \; n \tag{2}$$

$$x_i'(k) = x_i(k) + \sigma_i(k) N_j(0, 1) \tag{3}$$

Here r_j is a random number generated anew for each value of j. Its value can be from zero to any positive number. $x_i(k)$ and $x_i'(k)$ are the k-th component of $\vec{x_i}$ and $\vec{x_i}'$, respectively. $\sigma_i(k)$ is the standard

deviation for mutating the k-th component of $\vec{x_i}$. It is set as the average genotype distance between the k-th component of $\vec{x_i}$ and its ϕ strangers. $N_j(0,1)$ denotes a normally distributed one-dimensional random number with mean zero and standard deviation one for each value of j.

The essence of using modulo operation is that it may produce n or less than n distinct values for k depending on r_j. This arises two different scenarios for mutation. First, mutation changes all components of $\vec{x_i}$ one time when there are n distinct values for k. Second, when there are less than n distinct values, mutation does not change all components of $\vec{x_i}$ rather it changes some components one time and some components few times.

Step 7) Compute the fitness value of each offspring $\vec{x_i}'$, $i = 1, 2, \ldots, \mu$. Select μ individuals from parents and offspring for the next generation. If the fitness value of $\vec{x_i}'$ is at least equal to its parent $\vec{x_i}$, discard $\vec{x_i}$ and select $\vec{x_i}'$ for the next generation. Otherwise, discard $\vec{x_i}'$.

Step 8) If the best fitness value is acceptable or the maximum number of generations has been elapsed, stop the evolutionary process. Otherwise, continue.

Step 9) Repeat the following steps 10-13 for K_2 generations, which constitutes a single pass of the exploitation stage.

Step 10) For each individual $\vec{x_i}$, $i = 1, 2, \ldots, \mu$, select ϕ individuals from the population in such a way that the fitness value of them is very similar comparing to that of $\vec{x_i}$. The ϕ individuals, therefore, can be considered as *neighbors* i.e., nearest individuals for $\vec{x_i}$.

Step 11) Create μ offspring in the same way as described in step 6. However, the genotype distance of neighbors are used here instead of strangers used in step 6.

Step 12) Compute the fitness value of each offspring $\vec{x_i}'$, $i = 1, 2, \ldots, \mu$. Select μ individuals from parents and offspring for the next generation. If the fitness value of $\vec{x_i}'$ is better than its parent $\vec{x_i}$, discard $\vec{x_i}$ and select $\vec{x_i}'$ for the next generation. Otherwise, discard $\vec{x_i}'$.

Step 13) If the best fitness value is acceptable or the maximum number of generations has been elapsed, stop the evolutionary process. Otherwise, go to Step 4.

It is seen that RMEA uses the genotype distance of individuals as the standard deviation for mutation. The advantage of such an approach is that it makes the mutation operation self adaptive without using any adaptation scheme. The individuals in a population are spread over the entire search space at the beginning of an evolutionary process. As the evolutionary processes progresses, the population converges toward the optimal solution and the genotype distance between individuals reduces. This means mutation will explore a wider region of the search space at the beginning and a smaller region at the end of the evolutionary process. The necessary details of different stages and components of RMEA are given in the following subsections.

2.1 Exploration Stage

This stage facilitates to explore the wider region of a search space so that the chance of finding a good near-optimum solution by an evolutionary process is increased. RMEA uses the average genotype distance between an individual $\vec{x_i}$, that is going to be mutated, and the other ϕ individuals in the population as the standard deviation for mutation. The ϕ individuals are selected in such a way that they are very different with respect to $\vec{x_i}$ based on the fitness value. Since the fitness value of $\vec{x_i}$ and the ϕ individuals is very different, it is expected that their genotypes are likely to be very different. The population diversity tends to rise for using the average genotype distance of very different individuals and allowing offspring that have same fitness values as their parents for the next generation.

2.2 Exploitation Stage

It is quite natural to realize the potentials of already explored regions before further explorations. RMEA, therefore, executes several exploitation operations after exploration operations. The aim of exploitation stage is to reach the peaks of different explored regions so that the optimum solution, if exist in any peak, can easily be achieved. Like exploration stage, the same mutation is also used here. However, it differs from exploration stage in the way that the average genotype distance between an individual $\vec{x_i}$, that is going to be mutated, and its neighbors, is used as the standard deviation for mutation. It is expected that the the genotype of ϕ individuals and $\vec{x_i}$ is very similar resulting a small average distance i.e., standard deviation. This is beneficial for exploiting the local neighborhood because mutation produces offspring around parents.

2.3 Recurring Approach

It is known that executing exploration and exploitation operations separately, and combining them in one algorithm is beneficial for improving the performance of evolutionary algorithms. A number of approaches have been proposed in the literature that use GAs [6] for exploration and local search methods or specialized crossover operators for exploitation. According to [7], the following four issues must be addressed when exploration and exploitation operations are executed separately. First, when and where a local search method should be applied within the evolutionary cycle. Second, which individuals in the population should be improved by local search, and how they should be chosen. Third, how much computational effort should be allocated to each local search. Fourth, how genetic operators can be best integrated with local search in order to get a synergistic effect. It is here worth mentioning that the aforementioned four questions need also to be addressed even when the same method or operator is used for both exploration and exploitation.

To address these questions, a number of heuristics and of user specified parameters need to employ in any classical evolutionary algorithm. The employment

of many user specified parameters and heuristics requires a user to know rich prior knowledge, which often does not exist for complex real-world problems. Furthermore, it reduces the autonomy of an evolutionary process thereby may guide the process in the wrong direction resulting poor performance. A scheme that does not employ many heuristics and user specified parameters is clearly preferable. The repeated and alternated execution of exploration and exploitation operations on all individuals in a population could be the simple solutions for the first three questions, which is adopted in our proposed approach RMEA. The execution of exploration and exploitation operations on all the individuals is not problematic if the operations can be done adaptively by using the same evolutionary operator. Since RMEA uses only mutation for both exploration and exploitation operations, the fourth question is not arisen here.

3 Experimental Studies

The aim our experimental studies is to evaluate the performance of RMEA and to observe the effect of separating the exploration and exploitation operations in an evolutionary approach based on mutation. Both RMEA and CEP [3] are applied on six benchmark test functions. These are unimodal (f_1 and f_2), unrotated multimodal (f_3 and f_4), rotated multimodal (f_5) and composition (f_6) functions. The method described in [12] is used here to create the rotated function. It left multiples the variable x in the original function by the orthogonal matrix $\mathbf{M}$ to get a corresponding new variable y of the rotated function. The composition function CF3 proposed in [9] is used here. It is constructed by combining ten f_3s. The following is the the analytical forms of the six functions.

1) Sphere function: $f_1(x) = \sum_{i=1}^{D} x_i^2$
2) Schwefel's function: $f_2(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$
3) Griewanks's function: $f_3(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
4) Rastergin's function: $f_6(x) = \sum_{i=1}^{D} \left[x_i^2 - 10cos(2\pi x_i) + 10\right]$
5) Rotated Griewanks's function:
$f_5(x) = \frac{1}{4000} \sum_{i=1}^{D} y_i^2 - \prod_{i=1}^{D} \cos\left(\frac{y_i}{\sqrt{i}}\right) + 1 \qquad y = \mathbf{M} * x$
6) Composition function (CF3 [9]):
$f_6(x) = \sum_{i=1}^{10} \{w_i * [f_3((x - o_{inew} + o_{iold})/\lambda_i * M_i) + bias_i\}] + f_{bias}$

A. Experimental Setup

It is seen from section 2 that three user specified parameters K_1, K_2 and ϕ are used in RMEA. Among them, K_1 and K_2 are most important in the sense that they controls the behavior of RMEA. Three different sets of value are used to investigate the effect of K_1 and K_2. They are 1 and 1, 2 and 4, and 4 and 8. The value of ϕ is set 3 for all functions. The value of different parameters in function f_6 was set same as used in [9]. CEP [3] is implemented in this work according to [5].

Table 1. Performance of RMEA with different values for K_1, K_2 and CEP [4] on six different functions. The number of function evaluation and the dimension of functions were set 150,000 and 30, respectively.

	Mean best result for function					
	f_1	f_2	f_3	f_4	f_5	f_6
RMEA(1,1)	5.50e-017	1.44e-014	6.89e-014	3.59e-005	5.49e-014	1.36e-008
RMEA(2,4)	1.05e-017	2.21e-015	**6.41e-020**	1.47e-007	**7.99e-017**	**1.39e-010**
RMEA(4,8)	**9.44e-018**	**1.96e-015**	8.90e-020	**1.21e-007**	9.10e-017	3.25e-010
CEP	9.14e-004	2.16e+002	8.73e-002	4.37e+001	8.45e+001	7.66e-004

The population size μ was set 50 for both RMEA and CEP. The tournament size and the initial standard deviation used by CEP were set 5 and 3, respectively. The number of function evaluations (FEs) was set 150,000 for functions $f_1 - f_5$ and 50,000 for function f_6. The dimension of functions $f_1 - f_5$ was set 30, while it was set 10 for f_6. These values were chosen to make fair comparison with other work.

B. Results and Comparison

Table 1 shows the mean best result and standard deviation of RMEA and CEP on six functions over 50 independent runs. The numbers inside the parenthesis along RMEA indicate the values of K_1 and K_2 used in experiments. Fig. 1 shows the convergence characteristics of each function in terms of the mean best fitness.

It is clear that RMEA with different values for K_1 and K_2 performs much better than CEP [5]. The mean best fitness of RMEA with any value for K_1 and K_2 is better than CEP by several order magnitude. The t-test shows that the worst RMEA is significantly better than CEP for all six problems. The convergence characteristics of RMEA with different values for K_1,K_2 and CEP is similar at the very beginning of an evolutionary process (Fig. 1). As the evolutionary process progresses, the difference is very much clear. CEP appears to be trapped at the poor local optima or progresses very slowly. RMEA, on the other hand, successfully gets rid of local minima and progresses very aggressively toward a good near-optima. It is also clear from Table 1 that either RMEA(2,4) or RMEA(4,8) is better than RMEA(1,1). This indicates the necessity of executing exploration and exploitation operations repeatedly. The t-test shows that either RMEA(2,4) or RMEA(4,8) is significantly better than RMEA(1,1) for complex functions $f_3 - f_6$.

Tables 2 and 3 compare the performance of RMEA(2,4) with that of improved fast EP (IFEP) [14], adaptive EP with Lévy mutation (ALEP) [8] and real coded mematic algorithm (RCMA) with crossover hill climbing (XHC) [10]. To make the fair comparison, RMEA is reimplemented with the same number of function evaluation and problem dimension as used in RCMA with XHC. Like RMEA, both IFEP and ALEP use only mutation in producing offspring. IFEP

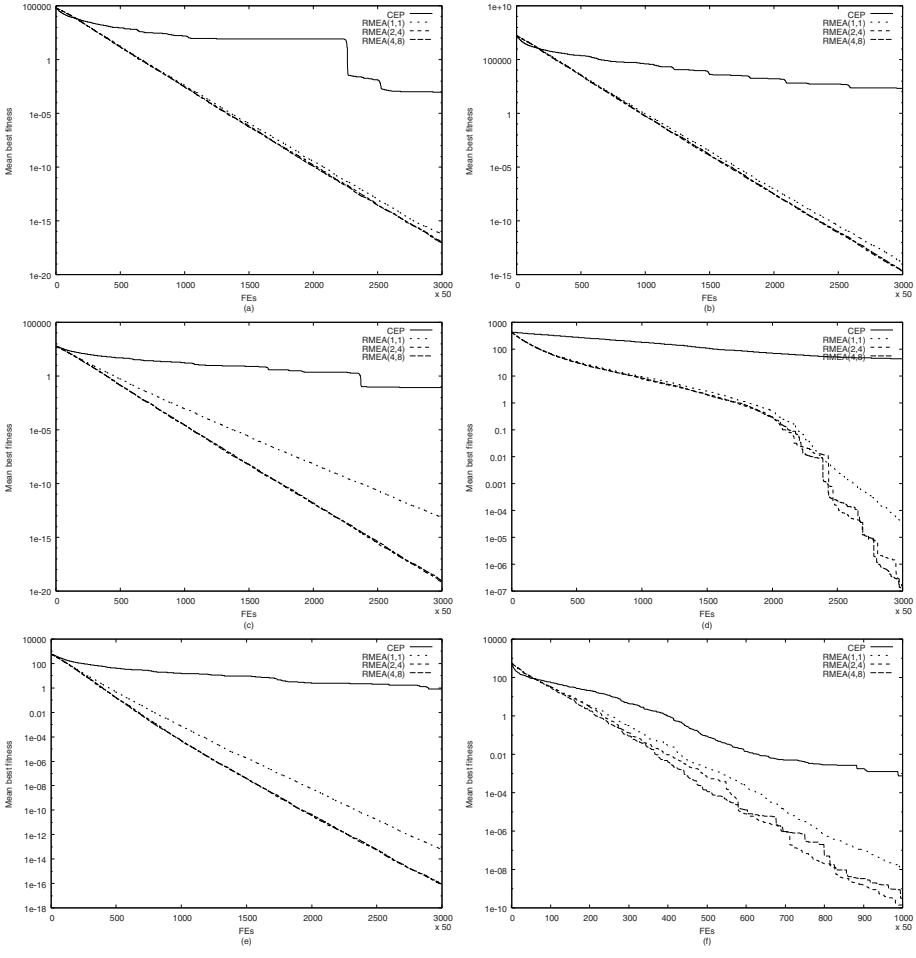


Fig. 1. Convergence characteristics of RMEA with different values for K_1, K_2 and CEP [4] on six functions: (a) f_1, (b)f_2, (c) f_3, (d) f_4, (e) f_5 and (f) f_6

Table 2. Comparison among RMEA, IFEP [14] and ALEP [8] on four functions. All results have been averaged over 50 independent runs. The number of function evaluation and the dimension of functions were set 150,000 and 30, respectively.

	Mean best result for function			
	f_1	f_2	f_3	f_4
RMEA(2,4)	**1.05e-017**	**2.21e-015**	**6.41e-020**	**1.47e-007**
IFEP	4.16e-005	-	4.53e-002	-
ALEP	6.32e-004	4.18e-002	2.4e-002	5.85e+000

Table 3. Comparison between RMEA and RCMA with XHC [10] on four functions. All results have been averaged over 50 independent runs. The number of function evaluation and the dimension of functions were set 100,000 and 25, respectively.

Algorithm	Mean best result for function			
	f_1	f_2	f_3	f_4
RMEA(2,4)	1.25e-013	**1.63e-011**	**1.08e-015**	**9.86e-004**
RCMA with XHC	**6.50e-101**	3.80e-007	1.3e-002	1.4e+000

mixes Cauchy and Gaussian mutations in one algorithm, while ALEP mixes Lévy mutations with four different distributions. RCMA with XHC executes exploration and exploitation operations separately. It uses PBX crossover [10] and BGA mutation [13] for exploration and a specialized crossover operator XHC [10] for exploitation.

It is clear from Table 2 that the performance of RMEA is better than IFEP and ALEP on all four functions we compared here. RCMA with XHC outperforms RMEA on one unimodal function (Table 3). However RMEA outperforms RCMA with XHC with one unimodal and two multimodal functions (Table 3). Although we could not perform t-test, the better performance of RCMA with XHC and RMEA seems to be significant.

4 Conclusions

RMEA introduces a recurring multi-stage framework for evolutionary algorithms in order to unravel the conflicting goals of exploitation and exploration during evolution. It has demonstrated very promising results, outshining some other algorithms on complex problems. Such an inspiring performance by RMEA is quite reasonable, because RMEA employs quite a different mechanism than the others. While most algorithms seem to stagnate during evolution especially at the late generation, RMEA still continues optimization process at a graceful rate. In fact, RMEA achieves log-linear convergence rate for all the six tested functions. This is because the alternating and repeating stages in RMEA ensure better immunity from stagnation. The principle characteristics of RMEA are controlled by three user-specified parameters K_1, K_2 and ϕ. Future work on RMEA includes making these parameters self-adaptive taking into account their effects on both fitness and diversity.

Acknowledgement. MMI is currently a Visiting Associate Professor at University of Fukui supported by the Fellowship from Japanese Society for Promotion of Science (JSPS). This work was in part supported by grants to KM from JSPS, Yazaki Memorial Foundation for Science and Technology, and University of Fukui.

References

1. Beyer, H.-G., Schwefel, H.-P.: Evolution Strategies: A Comprehensive Introduction. Natural Computing 1, 3–52 (2002)
2. Chellapilla, K.: Combining Mutation Operators in Evolutionary Programming. IEEE Transactions on Evolutionary Computation 2, 91–96 (1998)
3. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial Intelligence Through Simulated Evolution. Wiley, New York (1966)
4. Fogel, D.B.: Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. IEEE Press, New York (1995)
5. Gehlhaar, D.K., Fogel, D.B.: Tuning Evolutionary Programming for Conformationally Flexible Molecular Docking. In: Proc. of the Fifth Annual Conference on Evolutionary Programming, pp. 419–429. MIT Press, Cambridge, MA (1996)
6. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Kluwer Academic Publishers, Boston (1989)
7. Krasnogor, N., Smith, J.E.: A Tutorial for Competent Memetic Algorithms: Model, Taxonomy and Design Issues. IEEE Transactions on Evolutionary Computation 9, 474–488 (2005)
8. Lee, C., Yao, X.: Evolutionary Programming Using Mutations Based on the Lévy Probability Distribution. IEEE Transactions on Evolutionary Computation 8, 1–13 (2004)
9. Liang, J.J., Suganthan, P.N., Deb, K.: Novel Composition Test Functions for Numerical Global Optimization. In: Proc. of IEEE Swarm Intelligence Symposium, pp. 68–75 (2005)
10. Lozano, M., Herrera, F., Krasnogor, N., Molina, D.: Real-coded Memetic Algorithms with Crossover Hill-climbing. Evolutionary Computation 12, 273–302 (2004)
11. Moscato, P.: On Evolution, Search, Optimization, Genetic Algorithms and Martial arts: Towards Memetic Algorithms. Caltech Concurrent Computation Program, C3P Report 826 (1989)
12. Salomon, R.: Reevaluating Genetic Algorithm Performance Under Coordinate Rotation of Benchmark Functions. BioSystems 39, 263–278 (1996)
13. Schlierkamp-Voosen, D., Mühlenbein, H.: Strategy Adaptation by Competing Subpopulations. Parallel Problem Solving from Nature 3, 199–208 (1994)
14. Yao, X., Liu, Y., Lin, G.: Evolutionary Programming Made Faster. IEEE Transactions on Evolutionary Computation 3, 82–102 (1999)

The Effect of Missing Wind Speed Data on Wind Power Estimation

Fatih Onur Hocaoğlu and Mehmet Kurban

Anadolu University, Dept. of Electrical and Electronics Eng., Eskisehir, Turkey
{fohocaoglu,mkurban}@anadolu.edu.tr

Abstract. In this paper, the effect of possible missing data on wind power estimation is examined. One−month wind speed data obtained from wind and solar observation station which is constructed at Iki Eylul Campus of Anadolu University is used. A closed correlation is found between consecutive wind speed data that are collected for a period of 15 second. A very short time wind speed forecasting model is built by using two−input and one−output Adaptive Neuro Fuzzy Inference System (ANFIS). First, some randomly selected data from whole data are discarded. Second, 10%, 20% and 30% of all data which are randomly selected from a predefined interval (3−6 m/sec) are discarded and discarded data are forecasted. Finally, the data are fitted to Weibull distribution, Weibull distribution parameters are obtained and wind powers are estimated for all cases. The results show that the missing data has a significant effect on wind power estimation and must be taken into account in wind studies. Furthermore, it is concluded that ANFIS is a convenient tool for this kind of prediction.

1 Introduction

Wind is expected to be an important source of electric energy in the future in many regions. Many research groups in different countries have undertaken the development of commercial wind power plants. Wind speed is extremely important for electricity generation from wind turbine. The distribution of wind speeds is important for the design of wind farms, power generators. It is very important for the wind industry to be able to describe the variation of the wind speeds. The effective utilization of wind energy entails a detailed knowledge of the wind speed characteristics at a particular location. The characteristics of wind must be determined by using at least one year wind speed and wind direction data. The missing data should not exceed 10% according to the standards [1]. It is not possible to collect the data without any defect. There are a lot of studies for such numerical weather prediction problems in literature such as autoregressive moving average models (ARMA), Kalman filters [2], bilinear and smooth threshold autoregressive models, artificial intelligence techniques including the use of Multi Layered Perceptrons [3], Radial Basis Functions [4] and Recurrent Neural Networks [5] as well as Adaptive Neuro Fuzzy Systems. In [6] some of these models and different artificial intelligence based approaches

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 107–114, 2007.

are reviewed and compared in terms of Root Mean Square Error(RMSE) criteria for wind speed time series forecasting and better prediction results were obtained by ANFIS in most cases. The aim of this study is to minimize the effect of missing data on wind power estimation for a region which has not been taken into account in wind power estimation studies yet [7,8,9]. In this scope, one−month wind speed data with almost no defect that is measured and collected for a period of 15 seconds is studied. Some randomly selected data from whole data and from the interval (3−6)m/sec are discarded to represent missing data. According to high prediction capability at such studies an ANFIS model as described at Section 2 is built and the missing data are predicted for all cases. All data are fitted to the Weibull distribution mentioned in Section 3. The results are presented and discussed at Section 4 and 5 ,respectively.

2 ANFIS

ANFIS can incorporate fuzzy if−then rules and also, provide fine−tuning of the membership function according to a desired input−output data pair [10,11]. The ANFIS structure that is used for this study is given in Fig. 1.

In Fig. 1; first layer is known as input layer, each neuron in the second layer corresponds to a linguistic label and the output equals the membership function of this linguistic label, each node in layer 3 estimates the firing strength of a rule, which is found from the multiplication of the incoming signals, each node in layer 4 estimates the ratio of the i th rule's firing strength to sum of the firing strength of all rules, j, the output of layer 5 is the product of the previously found relative firing strength of the i th rule, the final layer computes the overall output as the summation of all incoming signals from layer 4. In this study weighted average procedure is used for defuzzify operation, a back−propagation training method is employed to find the optimum value for the parameters of MF(membership functions) and a least squares procedure is employed for the linear parameters on the fuzzy rules, in such a way as to minimize the error between the input and

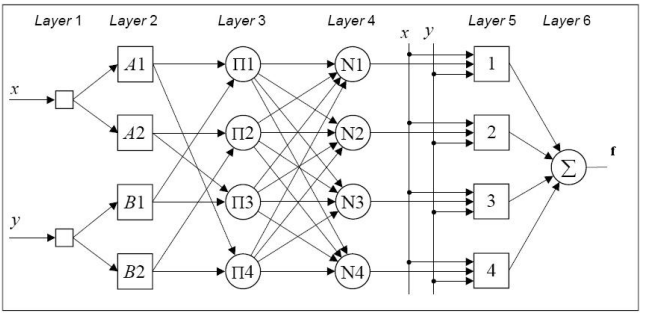


Fig. 1. The ANFIS structure

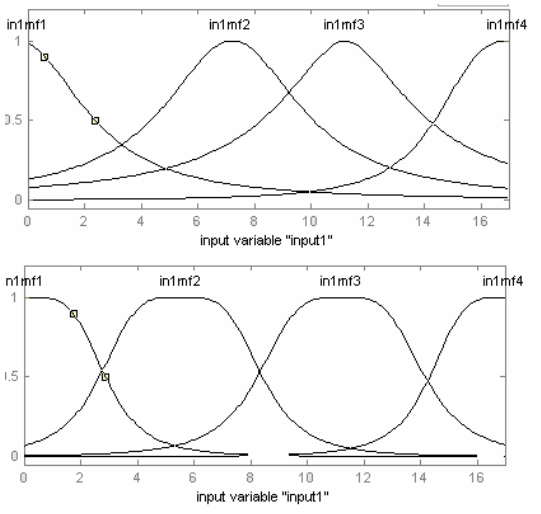


Fig. 2. The MFs of input1 before and after training, respectively

the output pairs, both input1 and input2 are divided into four MFs. The shapes of the MF's before and after training procedure are given in Fig. 2.

3 Weibull Distributions

To test the accuracy of the prediction operation the Weibull distribution which is the most popular due to its ability to fit most accurately the variety of wind speed data measured at different geographical locations in the world is used. Statistical estimation of unknown parameters from random sample is an important problem that can be solved by many establish methods, such as, the least square method (LSM), the weighted least square method (WLSM), the maximum likelihood method (MLM), the method of moments (MM), the method based on quantiles (QM) and a lot of modifications of these methods. The 2−parameter Weibull probability density function (p.d.f.) is given by equation. 1.

$$f_w(v) = (\frac{k}{c})(\frac{v}{c})^{k-1} \exp(-(\frac{v}{c})^k) \tag{1}$$

where $f_w(v)$ is the probability of observing wind speed v, k is shape parameters of Weibull p.d.f., c is scale parameters of Weibull p.d.f. the k values range from 1.5 to 3. for most wind conditions. The cumulative distribution of the Weibull distribution is given as follows:

$$F_w(v) = 1 - \exp(-(\frac{v}{c})^k) \tag{2}$$

Hennessey [12], Justus at al. [13], discussed a lot of estimation methods for Weibull distribution. In here, MLM is used. If $x_1, ..., x_n$ is random sample from

Weibull distribution, then the log−likelihood function, $L(c)$can be written as in equation. 3 and equation. 4 for MLM.

$$L(c,k) = \prod_{i=1}^{n} f_W(x_i, c) \tag{3}$$

$$L(c,k) = \prod_{i=1}^{n} (\frac{k}{c})(\frac{v}{c})^{k-1} \exp(-(\frac{v}{c})^k) \tag{4}$$

Taking the natural logarithm of likelihood function, we obtain equation, 5.

$$\ln L(c,k) = \sum_{i=1}^{n} \ln(\frac{k}{c}) + (k-1)\ln(\frac{k}{c}) - k\frac{v_i}{c} \tag{5}$$

any value c and k maximizing 5 is called maximum likelihood estimator (MLE), denoted by $\hat{k}_{MLE}$.

$$\frac{d\ln L(c)}{dc} = \sum_{i=1}^{n} \frac{-1}{c} + (k-1)(-\frac{1}{c}) + k\frac{v_i}{c^2} = 0 \tag{6}$$

$$\frac{d\ln L(c)}{dk} = \sum_{i=1}^{n} \frac{1}{k} + \ln(\frac{k}{c}) + (k-1)(\frac{1}{k}) - \frac{v_i}{c} = 0 \tag{7}$$

$\hat{k}_{MLE}$, $\hat{c}_{MLE}$ which maximizes 5 can be obtained from the solution of 6 and 7.

4 Results

The correlation coefficient between consecutive wind speeds are obtained to be 0.9857. The distributions of all data are given in Fig.3.

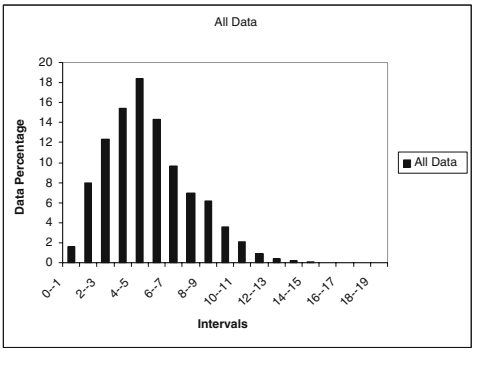


Fig. 3. The distributions of the data

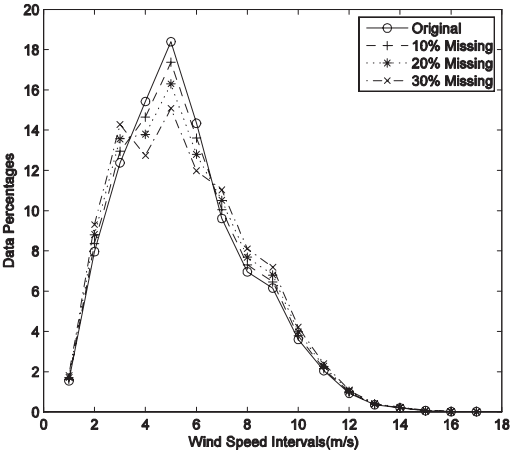


Fig. 4. The distributions after discarding

It is obtained that, when randomly selected data discarded from whole data the distributions did not change a lot and so power estimations from Weibull distribution did not change much more. From the interval 3−6 (m/s); 10%, 20% and 30% randomly selected data are discarded. The distributions of wind speed after discard are given in Fig. 4

It is obvious from Fig.5 that the distributions of the data are changed significantly. An ANFIS structure is built to predict the missing data. Two data sets, namely, the training and the testing set are employed for the ANFIS. Fig. 5 shows that the constructed ANFIS structure is convenient for the prediction operation.

The former data set is used during the identification process of the fuzzy model while the latter one is used to evaluate the forecast capabilities of the

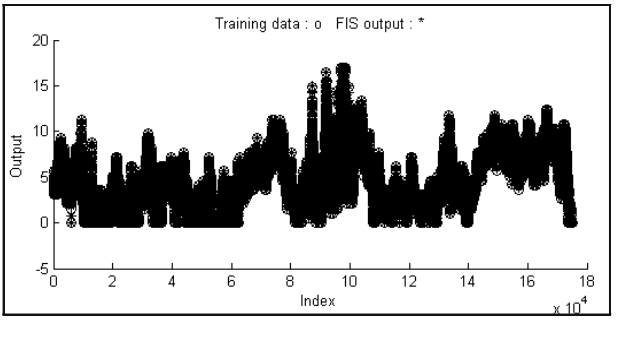


Fig. 5. The outputs that are found by ANFIS after training and actual data outputs

Table 1. The distributions after the prediction procedure

Wind speed(m/s)	Original	P10	P20	P30
0-1	1.5466	1.5466	1.5467	1.5466
1-2	7.9644	7.9644	7.9645	7.9644
2-3	12.3792	12.4144	12.4406	12.4721
3-4	15.4269	15.4761	15.5339	15.5417
4-5	18.394	18.3764	18.3948	18.4196
5-6	14.3402	14.2984	14.2148	14.1905
6-7	9.6117	9.5911	9.5748	9.5438
7-8	6.9590	6.9559	6.9530	6.9457
8-9	6.1508	6.1495	6.1502	6.1484
9-10	3.6011	3.6011	3.6012	3.6012
10-11	2.0509	2.0509	2.0509	2.0509
11-12	0.9296	0.9296	0.9296	0.9296
12-13	0.3556	0.3556	0.3556	0.3556
13-14	0.2021	0.2021	0.2021	0.2021
14-15	0.0607	0.0607	0.0607	0.0607
15-16	0.0121	0.0121	0.0121	0.0121
16-17	0.0152	0.0152	0.0146	0.0152

Table 2. Weibull distribution parameters and power estimations from Weibull distribution before and after data predictions

	$\hat{k}$	$\hat{c}$	$\hat{v}_m$	$\hat{\sigma}$	$\hat{v}_{mod}$	v_{max}
Original	5.5242	2.0331	4.8944	2.5209	3.9597	7.7373
10% Missing	5.5531	1.9985	4.9214	2.5742	3.9242	7.8567
After 10% pre.	5.5242	2.0335	4.8944	2.5205	3.9603	7.7365
20% Missing	5.5851	1.963	4.9516	2.6321	3.8857	7.9885
After 20% pre.	5.5095	2.0082	4.8824	2.5428	3.9093	7.7728
30% Missing	5.6211	1.9255	4.986	2.697	3.8422	8.1375
After 30% pre.	5.5237	2.0328	4.894	2.5211	3.9587	7.7373

obtained model. The data sets contain patterns formulated from historical data. The prediction values are incorporated to the discarded data and distributions are obtained as given in Table 1.

Where P10, P20 and P30 represent the distributions after incorporation of the predict of discarded 10%, 20% and 30% data into remaining data respectively. The values of the estimated powers after data prediction and the value of actual power are given on Fig.6. For all cases, the data are fitted to Weibull distribution, parameters are obtained and powers are estimated. The results are given in Table 2.

Finally, estimated wind powers after data prediction are obtained and given in Fig.6.

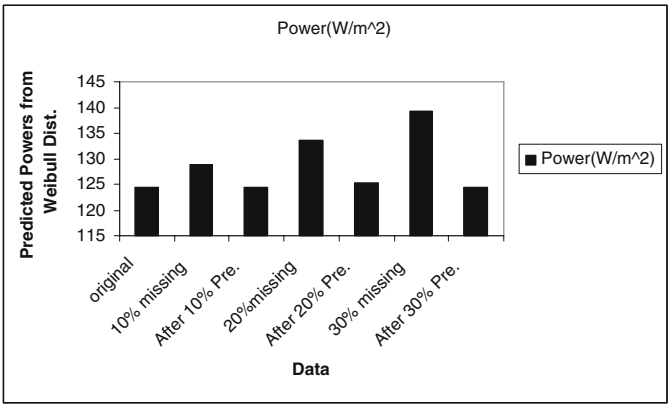


Fig. 6. The power values for actual data, missing data and forecasted data

5 Discussions and Conclusions

In this paper, the effect of missing data on determining the wind power estimation for a region is examined. It is thought that there can occur any fault in the data collection unit at random short time intervals. In this scope firstly, some randomly selected data are discarded from whole data. It is seen that the distributions of data did not change a lot. Then from the interval (3−6)m/sec randomly selected 10%, 20% and 30% of all data are discarded and it is observed that the distributions are changed significantly. It is obtained that if there are randomly occured faults they can be tolarated but if the faults are occured at specificaly time intervals in which the wind speed regime is in a specific interval that times it is necessary to consider the missing data effect on wind power estimation. In this paper, to consider the effect of the missing data an ANFIS structure is built. The missing data are predicted. The distributions of data are obtained by incorparating the missing data to remaining. Considering the Root Mean Square(RMS) Energy of the distribution of original data to be 38.1035, the RMS Errors between actual distribution and P10, P20, P30 are obtained to be 0.0190, 0.0436 and 0.0541, respectively. Also to test the accuracy of the model, the data are fitted to Weibull distribution and wind power estimations are obtained for all situations. The MLM is used to find the parameters of the Weibull distribution. Actual wind power per square meter is obtained to be 124 where the powers per square meters are obtained for the missing 10%, 20%and 30% data 129, 133 and 139 respectively. After incorporating the missing data to remaining, wind powers per square meter are obtained to be 124.8, 126 and 125.2, respectively. According to litrature, it is tolarable for 10% missing but it is concluded that if the missing data is sourced from a specific interval, the estimated wind power from Weibull distribution is obtained with 4.03% error. In conclussion, it is obvious from the results that, missing data has a significant effect on wind power estimation and must be taken into account in wind

power studies and ANFIS is a convernient tool for this kind of a study. Other known models such as latent variable modelling, bayesian estimation, bilinear and smooth threshold autoregressive models, kalman filters, ARMA models etc. may also be used to determine the effect of missing wind speed data together with ANFIS and best model may be established according to prediction results. Such studies can be regarded as a future work of this study.

References

1. Köse, R.: An evaluation of wind energy potential as a power generation source in Kütahya, Turkey. Energy Conversion and Management 45, 1631–1641 (2004)
2. Bossanyi, E.A.: Short-term wind prediction using Kalman filters. Wind Engineering 9, 1–8 (1985)
3. Alexiadis, M., Dokopoulos, P., Sahsamanoglou, H., Manousaridis, I.: Short term forecasting of wind speed and related electrical power. Solar Energy 63, 61–68 (1998)
4. Beyer, H.G, Denger, T., Hausmann, J., Hoffmann, M., Rujan, P.: Short term prediction of wind speed and power output of a wind turbine with neural networks. In: EWEC 1994. 5th European Wind Energy Association Conf, pp. 349–352 (1994)
5. Kariniotakis, G., Stavrakakis, G., Nogaret, E.: Wind power forecasting using advanced neural network models. IEEE Trans Energy Conversion 11, 762–767 (1996)
6. Sfetsos, A.: A comparison of various forecasting techniques applied to mean hourly wind speed time series. Renewable Energy 21, 23–35 (2000)
7. Ahmed Shata, A.S., Hanitsch, R.: The potential of electricity generation on the east coast of Red Sea in Egypt. Renewable Energy 31, 1597–1615 (2006)
8. Ozerdem, B., Turkeli, H.M.: Wind energy potential estimation and micrositting on Izmir Institute of Technology Campus, Turkey. Renewable Energy 30, 1623–1633 (2005)
9. Weisser, D.: A wind energy analysis of Grenada: an estimation using the Weibull density function. Renewable Energy 28, 1803–1812 (2003)
10. Jang, J.S.: ANFIS: Adaptive−Network−based Fuzzy Inference System. IEEE Trans. Systems, Man and Cybernetics 23, 665–684 (1993)
11. Jang, J.S., Sun, C.: Predicting Chaotic Time Series With Fuzzy If−Then Rules. In: Proc. IEEE Fuzzy Systems Conf., pp. 1079–1084 (1993)
12. Hennesessey, J.: Some Aspects of Wind Power Statistics. J. Appl. Meteoral 16, 119–128 (1977)
13. Justus, C., Hargraves, W.R., Mikhail, A., Graber, D.: Methods for estimating wind speed frequency distribution. J. Appl. Meteoral 17, 350–353 (1978)

Exploration of a Text Collection and Identification of Topics by Clustering

Antoine Naud[1,2] and Shiro Usui[1]

[1] RIKEN Brain Science Institute
2-1 Hirosawa, Wako City, 351-0198 Saitama, Japan
naud@brain.riken.jp, usuishiro@riken.jp
[2] Department of Informatics, N. Copernicus University
ul. Grudziadzka 5, 87-100 Torun, Poland

Abstract. An application of cluster analysis to identify topics in a collection of posters abstracts from the Society for Neuroscience (SfN) Annual Meeting in 2006 is presented. The topics were identified by selecting from the abstracts belonging to each cluster the terms with the highest scores using different ranking schemes. The ranking scheme based on log-entropy showed better performance in this task than other more classical TFIDF schemes. An evaluation of the extracted topics was performed by comparison with previously defined thematic categories for which titles are available, and after assigning each cluster to one dominant category. The results show that repeated bisecting k-means performs better than standard k-means.

1 Introduction

An increasing amount of published documents like research papers, computer programs, analyzed data or related references are gathered in databases or repositories in order to enable quick access to literature from a given field of research. The development of such databases in the field of neuroscience is a major goal in neuroinformatics [1]. The resulting large amounts of documents give rise to the need for tools that automatically organize them into indexing structures. These structures may fasten the retrieval for searched information as well as provide an overview of a corpus and help navigation. A subsequent task is the organization of the keywords in a structure reflecting the semantic contents of the documents. To this purpose, the general structure of a documents collection can be detected by clustering the documents into groups covering similar topics. This work is devoted to the analysis of the posters presented at the Annual Meeting of the Society for Neuroscience (SfN) in 2006. SfN is, with more than $37,500$ members, the world's largest organization of scientists devoted to the study of neuroscience and the brain science. Its Annual Meeting is the largest event in neuroscience. The primary goal of this work was the automatic discovery of topics covered in poster sessions, on the basis of the posters abstracts and titles. Another potential application is the automatic partitioning into sessions of the posters submitted to future SfN Annual Meetings.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 115–124, 2007.

2 Construction of the Vector Space Model

The most widely used approach in Natural Language Processing is the *vector space model*. In this model, a set of terms $\mathcal{T}$ is first built by extracting all words occurring in a collection of documents $\mathcal{D}$, followed by stop words removal and stemming steps [2]. The number of occurrences of each term in each document (usually called *frequency*) is counted and denoted f_{ij}. Then a frequency matrix $\mathbf{F}$ is built with the $\{f_{ij}\}$ as entries. As we will cluster documents in this work, it is more convenient to build $\mathbf{F}$ as a $[documents \times terms]$ matrix, where each document is a row vector in the space of all terms, called the *term space* later on. Depending on the purpose of the application, terms occurring too often or very seldom can also be discarded. When the number of documents N in the collection is in the range of a few thousands, the number of extracted terms M is often larger than a few tens of thousands, leading to very high dimensional space for the documents. In order to remove less semantically significant terms and also to enable further processing, it is necessary to reduce the term space dimension by selecting a smaller subset of terms, usually using a ranking of the terms according to their Document Frequency (DF). In general, we are interested in selecting the terms that best represent the semantic content of the documents. This intuitive feature is however very difficult to catch only by statistical means. In the present application, the terms were extracted from the posters' abstracts and titles. The preprocessing scheme and extraction of candidate terms was the same as in [3]. From the abstracts and titles of the $N = 12844$ posters, we obtained directly $M = 40767$ terms, which is a too large value to allow further processing. 3 term spaces were built by selecting terms occurring in at least $2, 13$ and 45 documents for the following reasons: (a) selecting terms with $DF \geq 2$ allows to decrease the term space size roughly by a factor of two, leading to $M = 19794$ terms; (b) selecting terms with $DF \geq 13$ leads to $M = 6127$, this is the maximal size allowing the application of Matlab's kmeans function in section 5, (c) selecting terms with $DF \geq 45$ decreases again by two the number of terms, ending up with $M = 3006$ terms. Only unigrams (single words) were considered for the terms in this preliminary study.

3 Exploratory Analysis of Existing Categories

The posters abstracts and titles were extracted from a CD-ROM distributed to all the participants of the Annual Meeting. Four types of categories are provided by the Meeting's organizing committee: *theme*, *subtheme*, *topic* and *session*, and a name is given to each category. Each of the 12844 posters for which an abstract and a title (called hereafter *documents*) were available was also assigned by the organizers to one poster session, one topic, subtheme and theme. A summary of basic statistics of this collection of documents is given in Table 1. The purpose of this analysis is to check whether the various originally defined groupings of posters into categories can be observed in the term spaces that we defined in the previous section.

Table 1. Summary data of the Society for Neuroscience 2006 Annual Meeting

1	Number of themes	7
2	Number of subthemes	71
3	Number of topics	415
4	Number of poster sessions	650
5	Number of poster abstracts	12844
6	Number of words / abstract (average)	278
7	Number of extracted terms	40767

3.1 Average Cosine Measures Between Documents

The frequency matrix $\mathbf{F}$ is a sparse contingency table where each row represents one document, and the similarity of two documents can be evaluated by the cosine of the angle between the two document vectors. In order to balance the frequencies of terms occurring in long abstracts with respect to terms occurring in shorter abstracts, a normalization of the rows of matrix $\mathbf{F}$ is performed after the term weighting (see [4] for a review of weighting schemes). The cosine between 2 vectors in the high-dimensional term space is defined as

$$cos(\mathbf{d}_1, \mathbf{d}_2) = \frac{\mathbf{d}_1 \cdot \mathbf{d}_2}{\|\mathbf{d}_1\| \|\mathbf{d}_2\|}, \tag{1}$$

where $\cdot$ is the dot product. As vectors $\{\mathbf{d}_i\}$ are of unit length, expression (1) simplifies to the dot product. The mean cosine for all pairs of documents within each category is a measure of how dense are the categories in the term space. Similarly, for each category, the mean of the cosines between each document in the category and all the documents in all other categories measures to which extend this category is separated from the others. The averages of these two means for all the categories were computed efficiently using the centroid vectors of each category, as described in [5]. The results are presented in Figure 1. Note that the cosine function is a similarity measure (i.e. the more similar two documents are, the higher is their cosine) and not a distance (or dissimilarity). The average cosines within categories are clearly higher than between categories in each term space, especially for the *topic* and *session* categories, which indicates that these categories are also well defined in the 3 term spaces. The above two average cosines among categories are equivalent to clusters' *cohesion* and *separation*, some internal measures of clusters validity presented e.g. in [6].

3.2 MDS Layouts of the Original Categories

As it was seen above, the differences in average cosines between and within categories are larger for *topic* and *session* categories, which indicates that those categories are better separated in the terms space. This can be confirmed by visualizing the different categories. To this purpose, we processed the data as follows:

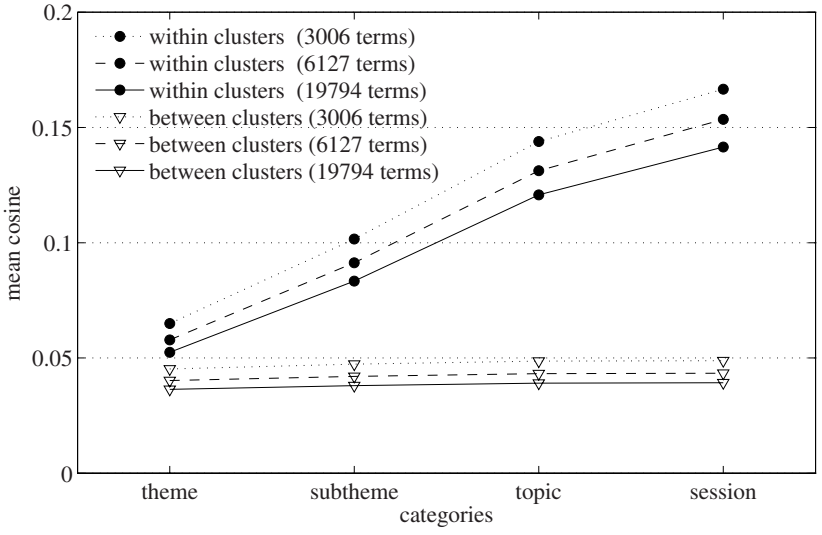
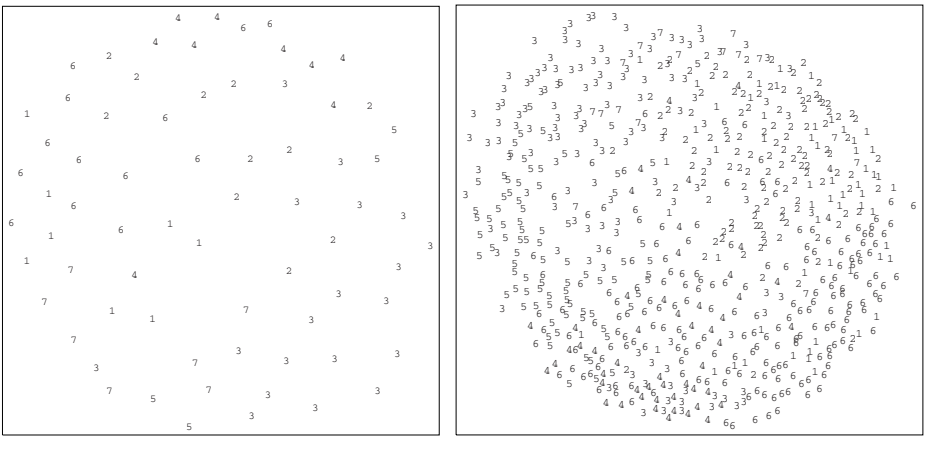


Fig. 1. Mean cosines among original categories in the 3006, 6127 and 19794 term spaces

1. Build a similarity matrix C with mean cosines between categories as entry and mean cosines within categories on its diagonal,
2. Compute a dissimilarity matrix $D = -log(C)$, in order to have squared distance measures instead of similarities,



(a) 71 *subtheme* categories **(b)** 650 *session* categories

Fig. 2. MDS layouts of original categories in the 3006 terms space. The different numbers represent the dominant themes in each category.

3. Map the categories using multidimensional scaling (MDS) [7] or Spherical Embedding algorithm [8] (using the dissimilarity matrix D as input distances) into a 2-D or 3-D space. [1]
4. Plot the 2-dimensional layout of categories, marked according to the dominant theme, that is the theme, which has the largest number (majority) of abstracts among all the abstracts belonging to that category.

The layouts resulting from least squares MDS mapping of 2 types of categories (*subtheme* and *session*) are presented in Figure 2. We observe that the items of these 2 types of categories are mapped in good agreement with the *theme* categories because their marks are clustered. This also confirms the conclusion of section 3.1.

4 Identification of Documents Subsets

4.1 Proposed Approach for Topic Identification

We assume that documents belonging to a given category refer to a common topic. The topics of the categories are naturally best described by their given titles, so we just wanted to check to what extend are we able to retrieve these titles. The topic of a set of documents was identified by extracting the most important terms occurring in these documents. To this purpose, 3 ranking schemes were used: a) the Document Frequency (denoted hereafter DF), b) the Term Frequency-Inverse Document Frequency, or TF-IDF (hereafter TI), c) the Log-Entropy (hereafter LE). They are defined for each term $t_j, j = 1, ..., M$ as follows:

$$\begin{aligned} DF\,(t_j) &= \sum_{i=1}^{N} \chi\,(f_{ij}), \quad \text{with } \chi(t) = 1 \text{ if } t > 0 \text{ and } \chi(0) = 0 \\ TI\,(t_j) &= \sum_{i=1}^{N} f_{ij}\ .\ \log\left(\frac{N}{\sum_{i=1}^{N} \chi(f_{ij})}\right), \\ LE\,(t_j) &= \sum_{i=1}^{N} \log\,(1 + f_{ij})\ .\ \left(1 + \sum_{i=1}^{N} \frac{p_{ij} \log p_{ij}}{\log N}\right), \quad \text{with } p_{ij} = f_{ij} / \sum_{i=1}^{N} f_{ij} \end{aligned} \tag{2}$$

For each type of category, the top 20 terms were selected using the 3 rankings defined above, in the 3 term spaces built in section 2. The numbers of terms (among the top 20 ranked or all the terms) matching after stemming one term of the category title were counted. Table 2 presents the results. We get naturally the best possible results when taking all the terms (*NO* ranking) extracted from the abstracts. We can see that the log-entropy ranking (LE) performs the best among the 3 rankings, with an average retrieval score of 54.0 % (against 53.1 % for DF and 13.0 % for TI). Another result is that there is no significant decrease of performance when the term space size k decreases, which means that the strategy based on Document Frequency for building the terms space is sensible.

[1] MDS was used rather than PCA because the feature matrix $\mathbf{F}$ is too large to allow its direct decomposition by the classical (non-sparse) versions of PCA calculations.

Table 2. Numbers of retrieved terms of the categories titles among the top 20 terms using different rankings (*TI, DF, LE*) or among all terms (*NO* ranking). The percentages in parenthesis are calculated wrt the numbers of title terms in the fourth column.

M	Category titles			Top 20 terms rankings						All terms	
	name	(# cat.)	# terms	*DF*	(%)	*TI*	(%)	*LE*	(%)	*NO*	(%)
3006	*theme*	(7)	16	3	(18.8)	3	(18.8)	**4**	(25.0)	15	(93.7)
	subtheme	(71)	168	87	(51.8)	43	(25.6)	**88**	(52.4)	151	(89.9)
	topic	(415)	1111	606	(54.5)	163	(14.7)	**610**	(54.9)	976	(87.8)
	session	(650)	2191	1138	(51.9)	289	(13.2)	**1163**	(53.1)	1883	(85.9)
6127	*theme*	(7)	16	3	(18.8)	3	(18.8)	**4**	(25.0)	15	(93.7)
	subtheme	(71)	168	89	(53.0)	40	(23.8)	**90**	(53.6)	158	(94.0)
	topic	(415)	1111	615	(55.4)	154	(13.9)	**619**	(55.7)	1022	(92.0)
	session	(650)	2191	1152	(52.6)	256	(11.7)	**1179**	(53.8)	1968	(89.8)
19794	*theme*	(7)	16	3	(18.8)	3	(18.8)	**4**	(25.0)	15	(93.7)
	subtheme	(71)	168	89	(53.0)	39	(23.2)	**90**	(53.6)	160	(95.2)
	topic	(415)	1111	615	(55.4)	141	(12.7)	**617**	(55.5)	1041	(93.7)
	session	(650)	2191	1153	(52.6)	222	(10.1)	**1179**	(53.8)	2000	(91.3)

4.2 Identified Topics for the Original Categories

Table 3 presents a list of the 10 first session titles for which all title terms are among the top 20 log-entropy ranked terms, extracted from the posters' titles and abstracts belonging to this session. There were 130 entirely retrieved titles among the 650 sessions.

5 Clustering of the Abstracts and Evaluation

5.1 Clustering Experiments

The primary rationale for clustering the abstracts is to try to build the different thematic categories in an automatic manner. For this reason, and to allow a comparison with the original categories, the abstracts were clustered into k clusters, for $k = 7, 71, 415$ and 650. Among the numerous existing clustering algorithms, we chose k-means for this analysis, because it was reported to perform well on documents [5]. K-means was used in two versions: (i) standard (naive) k-means and (ii) bisecting k-means (or *repeated bisections*) introduced in [5]. The k-means algorithm has been successfully applied to cluster large collections of documents as it scales relatively well with the space dimensionality, especially when the cosine similarity is used and the vectors are normalized [9], in the so-called *spherical k-means*. Matlab `kmeans` function with cosine distance measure was used as spherical k-means, and the repeated bisections k-means used was the `vcluster` function (with default parameters) from CLUTO clustering package [10]. In a purpose of comparing these two versions of k-means clustering, the clusters resulting from both functions have been evaluated by comparison with previously defined classes, namely the thematic categories provided by the

Table 3. 10 session titles with the selected terms in the 3006 term space. Boldface terms matched one title word after stop word removal and stemming. Title words like *and, other, neural* or numbers are in the stop list.

Session title	Top 20 terms (log-entropy ranking)
Serotonin Receptors I	**receptors** HT proteins rats functional agonist signals antagonist **serotonin** regulation Inhibition drugs brain dose injecting pathway assay coupled OH DPAT
Ion Channels: Trafficking and Other	**channel** proteins membrane subunits functional **ions** regulation interaction voltage form hippocampal domains gating dendritic cultured potentials conductance local surface **trafficking**
Dopamine Transporters I	DAT **transport dopamine** DA regulation proteins uptake functional phosphorylated surface terminal interaction synaptic internal site Inhibition cocaine membrane trafficking kinase
Short-Term Plasticity	synaptic potentials synapse presynaptic action depolarized recordings **short** release **term** regulation **plasticity** Layer cortical Inhibition vesicle trains amplitude form transmission
LTD I	**LTD** receptors synaptic depressant mGluRs hippocampal long term CA1 proteins form plasticity stimulation synapse NMDAR AMPA glutamate DHPG required AMPAR
Neural Oscillators	membrane potentials intrinsic **oscillation** spike models dynamics depolarized recordings properties voltage hyperpolarizing channel synaptic mV clamp conductance thresholding slowing low
Retina I	**retinal** light photoreceptors functional visual recordings mice bipolar rods proteins processes cones Dark synapse determined membrane receptors degeneration rats synaptic
Retina II	**retinal** ganglion receptors functional RGCs light pathway ON Layer visual recordings dendritic stimulus properties signals mice stimulation modulation field photoreceptors
Eye Movements: Saccades	**saccadic eye** monkey stimulus fixating visual **movements** error direct anti located field instructed pro cue reaction SC points signals Inhibition
Trigeminal Processing	**trigeminal** rats pain injecting receptors nociception regions modulation behavioral stimulation chronic central ganglion formalin nucleus **processes** hyperalgesia sensitive sensory spinal

meeting's organizers. We used the following external measures of clusters validity: *purity*, *entropy*, *F-measure* and *Mutual Information*, as proposed in [11]. These measures assess to which extend two objects from the same class (category) are in the same cluster and vice-versa. Table 4 summarizes the evaluation of clusters obtained by standard and repeated bisections k-means in 3006 and 6127 term spaces, clustering in the 19794 term space was not performed due to excessive memory requirements. It can be observed that repeated bisecting k-means algorithm performs better in terms of Entropy and Mutual Information, whereas spherical k-means is better in terms of Purity and F-measure. Relying primarily on Mutual Information, which is a theoretically well founded and unbiased measure, we conclude that our experiments confirm that repeated bisection performs better than spherical k-means, as reported in [5]. For both of the applied clustering techniques, the quality of the clusters increases with a decreasing k, indicating that categories *theme* and *subtheme* correspond in these term spaces to real clusters in a better way than *topic* and *session* categories. The results are slightly better in the 6127 term space in terms of Purity,

Table 4. External measures of cluster validity for the clusterings obtained from spherical k-means and repeated bisecting k-means. An up arrow ↑ (resp. down arrow ↓) below the measure name indicates that a higher (resp. lower) value means a better clustering. Boldface entries identify the best result according to each measure, for each (M, k) pair.

clustering algorithm	M	k	Purity ↑	Entropy ↓	F-measure ↑	Mut. Inf. ↓
spherical k-means	3006	7	**0.543**	0.344	**0.486**	0.251
		71	**0.441**	0.510	**0.359**	0.404
		415	**0.285**	0.608	**0.253**	0.559
		650	**0.240**	0.641	**0.242**	0.635
	6127	7	**0.565**	0.363	**0.517**	0.270
		71	**0.448**	0.512	**0.363**	0.407
		415	**0.299**	0.617	**0.266**	0.568
		650	**0.255**	0.648	**0.252**	0.642
repeated bisecting k-means	3006	7	0.505	**0.300**	0.427	**0.207**
		71	0.380	**0.459**	0.302	**0.353**
		415	0.248	**0.578**	0.216	**0.528**
		650	0.206	**0.612**	0.207	**0.606**
	6127	7	0.507	**0.301**	0.434	**0.208**
		71	0.384	**0.464**	0.298	**0.359**
		415	0.253	**0.581**	0.219	**0.532**
		650	0.210	**0.615**	0.209	**0.609**

whereas the 3006 term space performs better in terms of Entropy and Mutual Information, this last term space having a lower amount of 'noisy' terms.

5.2 Identification of Topics for the Clusters

Once we have performed the clustering of the documents, we extracted terms from the abstracts of each obtained cluster in a similar manner as in section 4.1, in order to identify the topics covered by the clusters. We selected again the top 20 terms according to a log-entropy ranking of the terms occurring in the cluster's documents. Finally, we assigned each cluster to one original category, in order to check the selected terms against the category's title (for $k = 7$ clusters, we assigned each cluster to one of the 7 themes, for $k = 71$, we assigned to one of the 71 subthemes, and so on...). The assignment was done to the *dominant* category: For all the documents in a cluster, the original categories of the documents were counted (we built the histogram of the categories) and the cluster was assigned to the category for which the number of documents was the largest. The top 10 terms, according to the LE ranking, were selected in the 3006 and 6127 term spaces. The numbers of retrieved title terms of the assigned categories is expectedly lower than for the original categories (we select only 10 terms instead of 20 and we don't use the original categories defined by human experts), but still satisfying with an average of 32.1% retrieved title terms in the

Table 5. Selected terms identifying topics of 10 clusters among the 66 category titles entirely retrieved (out of the 415 *topic* categories) in the 3006 terms space

Assigned title	Top 20 terms (log-entropy ranking)
Maternal behavior	**maternal behavioral** pups rats care offspring lactate mothers mice receptors
Opioid receptors	morphine **opioid receptors** tolerance rats mice analgesia injecting analgesic dose
Motor unit	muscle contract Forced **motor** isometric voluntary **unit** EMG rate variables
Aggression	**aggression** behavioral social mice Intruder receptors brain models rats Resident
Alcohol	ethanol rats **alcohol** intake consumption receptors drinking behavioral water dose
Metabotropic glutamate receptors	mGluRs **receptors glutamate metabotropic** III rats synaptic mGluR5 synapse regulation
Reward	NAc rats accumbens nucleus behavioral DA **reward** drugs dopamine shell
Cocaine	**cocaine** drugs exposure rats receptors brain behavioral abstinence withdrawal regions
Transplantation	grafting rats **transplants** axonal regenerate cord nerves Survival spinal injury
Parkinson's disease Models	MPTP mice **Parkinson disease models** PD DA dopamine dopaminergic striatal

3006 term space, and 34.0% in the 6127 terms space. This demonstrates that the k-means approach is well suited to this practical application. As an illustration, a list of top 10 terms for 10 clusters (for which all the assigned title's terms were retrieved) obtained by repeated bisections with $k = 415$ is presented in Table 5. Boldface terms matched, after stemming, one word from the assigned category title.

6 Conclusions

This preliminary analysis of abstracts of posters presented at SfN 2006 Annual Meeting shows that the original thematic categories are to some extend separated in the term spaces extracted from posters abstracts and titles: it was possible to extract from the documents 54.0% of all the titles words of these categories. The log-entropy ranking scheme performed better than TF-IDF or DF rankings. A clustering of the abstracts using two versions of k-means algorithm resulted in clusters of higher average quality for repeated bisections in terms of Entropy and Mutual Information. An identification of topics, performed by selection of terms from the abstracts was also performed. Each of the obtained clusters was assigned to one original thematic categories by choosing the category with the majority of abstracts. These clusters were also evaluated in terms of their capacity to retrieve their assigned category titles. The achieved performance is satisfying as compared to the retrieval rates for original categories. The results can be further improved, e.g. by applying more elaborate methods for the selection of relevant terms, in particular by using bigrams. By construction, k-means algorithms assume that the clusters are spherical and of similar densities, which might be untrue in the case of documents. An effort towards finding clustering techniques that are better suited to documents collections is noticeable in the

literature, among others based on Nonnegative Matrix Factorization. A comparison of these techniques with the approach adopted in the present research is envisaged.

Acknowledgments

The authors wish to thank to the Society for Neuroscience for granting them the use of SfN 2006 Annual Meeting abstracts. Discussion with N. Ueda from NTT-CS, Kyoto, Japan and a collaboration on the Vector Space Model construction with T. Taniguchi from IVIS Inc., Tokyo, Japan are gratefully acknowledged.

References

1. Usui, S.: Visiome: Neuroinformatics Research in Vision Project. Neural Networks 16, 1293–1300 (2003)
2. Porter, M.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
3. Usui, S., Palmes, P., Nagata, K., Taniguchi, T., Ueda, N.: Keyword Extraction, Ranking, and Organization for the Neuroinformatics Platform. Bio Systems 88, 334–342 (2007)
4. Kolda, T.G.: Limited-memory matrix methods with applications. University of Maryland, CS-TR-3806, ch. 7, pp. 59–78 (1997)
5. Steinbach, M., Karypis, G., Kumar, V.: A comparison of documents clustering techniques. In: KDD Workshop on Text Mining (2000)
6. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to datamining. Addison-Wesley, Reading (2006)
7. Groenen, P.: Modern multidimensional scaling: Theory and Applications. Springer Series in Statistics. Springer, Heidelberg (1996)
8. Saito, K., Iwata, T., Ueda, N.: Visualization of Bipartite Graph by Spherical Embedding. In: JNNS (in Japanese) (2004)
9. Dhillon, I.S., Modha, D.S.: Concept decomposition for large sparse text data using clustering. Machine Learning 42(1/2), 143–175 (2001)
10. CLUTO, Karypis, G., et al.: University of Minnesota (2003), available at: http://glaros.dtc.umn.edu/gkhome/views/cluto
11. Strehl, A., Ghosh, J., Mooney, R.: Impact of similarity measures on Web-page clustering. In: AAAI 2000. Proc. AAAI Workshop on AI for Web Search, Austin, pp. 58–64. AAAI-MIT Press (2000)

Asynchronous BCI Control of a Robot Simulator with Supervised Online Training

Chun Sing Louis Tsui and John Q. Gan

BCI Group, Department of Computer Science,
University of Essex, Colchester, CO4 3SQ, United Kingdom
{csltsu, jqgan}@essex.ac.uk

Abstract. Due to the non-stationarity of EEG signals, online training and adaptation is essential to EEG based brain-computer interface (BCI) systems. Asynchronous BCI offers more natural human-machine interaction, but it is a great challenge to train and adapt an asynchronous BCI online because the user's control intention and timing are usually unknown. This paper proposes a novel motor imagery based asynchronous BCI for controlling a simulated robot in a specifically designed environment which is able to provide user's control intention and timing during online experiments, so that online training and adaptation of motor imagery based asynchronous BCI can be effectively investigated. This paper also proposes an online training method, attempting to automate the process of finding the optimal parameter values of the BCI system to deal with non-stationary EEG signals**.** Experimental results have shown that the proposed method for online training of asynchronous BCI significantly improves the performance.

Keywords: Adaptation, asynchronous BCI, brain-computer interface, online training, automated learning.

1 Introduction

A brain-computer interface (BCI) is a communication system in which an individual sends commands to the external world by generating specific patterns in brain signals and a computer detects and translates the brain signal patterns into commands that accomplish the individual's intention. BCI systems can be categorised into systems operating in cue-paced (synchronous) or self-paced (asynchronous) mode. The majority of the existing EEG-based BCI systems are synchronous [1]-[3], in which the analysis and classification of brain signals is locked to predefined time windows. This means that users are supposed to generate commands only during specific periods determined by the BCI system. The advantage of synchronous BCI systems is that the onset of mental activity is known in advance and associated with a specific cue or trigger stimulus, and thus any signal outside the predefined time windows is treated as idling and ignored by the BCI system. On the other hand, asynchronous BCI systems offer a more natural mode of human-machine interaction than synchronous BCIs [4]-[9]. In asynchronous BCIs, no cue stimulus is used, the users

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 125–134, 2007.

control the BCI output whenever they want by intentionally performing a specific mental/cognitive task, and the EEG signals have to be analysed and classified continuously. The disadvantage of asynchronous BCI is that the lack of indications of the user's control intention and timing brings about challenges in asynchronous BCI design and performance evaluation.

It is well-known that EEG signals, particularly in EEG-based BCI systems, are non-stationary. The non-stationarities may be caused by the subject's brain conditions or dynamically changing environments. To some extent, a realistic BCI system has to be trained online and adaptive even in application phases where the true labels of ongoing EEG trials are unknown [10]-[17]. It is a great challenge to train and adapt an asynchronous BCI online because the user's control intention and timing are usually unknown to the BCI system. Existing methods for obtaining the user's control intention and timing in asynchronous BCI are to analyse continuous EEG data consisting of defined cue-triggered mental states (simulated asynchronous BCI) [4], use self-report by subjects [5][6], rely on subjects to perform a predefined sequence of mental tasks [5][7], or analyse real movement EEG data recorded along with time stamp of actual movements [8][9]. It should be noted that the above methods are not so suitable for online training purpose, except for the latter approach which assumes that the imagined movements will have enough similarity to actual movements. However, realistic BCI applications should be based on motor imagery or other mental activities rather than real movements.

This paper proposes a novel motor imagery based asynchronous BCI control of a simulated robot in a specifically designed environment, which is able to provide user's control intention and timing during online experiments, so that online training and adaptation of motor imagery based asynchronous BCI can be effectively investigated. This paper also proposes an online training method, attempting to automate the process of finding the optimal parameter values of the BCI system to deal with non-stationary EEG signals. Experiments have been conducted, producing promising results on performance improvement of asynchronous BCI by online training in terms of accuracy and deviation of real paths from optimal paths.

2 Methods

2.1 BCI Experiment Setup

The experiments were carried out with able-bodied subjects who sat on an armchair at 1m distance in front of a computer screen. The EEG recording was made with a g.tec amplifier (Guger Technologies OEG Austria). Five bipolar EEG channels using 10 electrodes, as shown in Fig. 1, were measured over C3 (FC3 vs. CP3), C1 (FC1 vs. CP1), Cz (FCz vs. CPz), C2 (FC2 vs. CP2), and C4 (FC4 vs. CP4). The EEG was sampled at 250Hz.

2.2 Offline Training

A simple synchronous BCI paradigm, proposed by the Graz BCI Lab [3], was used to record data for training classifiers offline before online experiments. The subjects were asked to imagine left versus right hand movements. The experiment consisted of

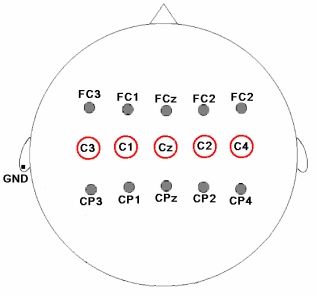


Fig. 1. Five bipolar electrode positions

6 runs with 40 trials each. In each trial the subjects relaxed until a green cross appeared on screen at t = 2s (s for second). At t=3s, a red arrow (cue) pointing either left or right direction appeared on screen for 2 seconds. The subject's task was to respond to the arrow by imagine left or right hand movements until the green cross disappeared at t = 8s. The order of left and right cues was random, and there was a random interval of 2~3 seconds between trials.

Logarithmic band power features were extracted from EEG signals and used to classify the imagery movements into left or right class. Two frequency bands that give good separation were manually selected for each subject. Using the selected frequency bands, EEG signals were digitally bandpass filtered, squared, averaged over a 1 second sliding window, and a natural logarithm was then applied to obtain the features. Using the extracted features and their corresponding class labels (from the cue signals), two linear discriminant analysis (LDA) classifiers were trained, with one to distinguish left imagery movement from others (right imagery movement or no imagery movement) and the other to separate right imagery movement from others.

It was shown in the BCI competition 2003 and 2005 that LDA performs as well as (sometimes even outperforms) non-linear classifier, and almost all the winning classifiers are linear [18]. Therefore, we chose to use LDA in our design.

2.3 Online Asynchronous Event Detection

During online asynchronous BCI experiments, the extracted features, which are related to the user's control intent, were continuously classified by the offline trained LDA classifiers and used to control a robot simulator that is described in detail in section 2.4. The online asynchronous event detection system used in the experiments is shown in Fig. 2, which works as follows. LDA outputs below a threshold will be set to zero. If a LDA output, used as a class's confidence value, is above the threshold long enough (>Dwell_Length), the dwell requirement of an event onset is met, a class (either left or right) will be selected as a command to control the simulated robot, and a refractory period will be switched on at the same time to reject new class/action to be triggered until the refractory period ends. The threshold and dwell mechanism plus refractory period are effective methods for reducing false positive rate. Refractory period also introduce a competition mechanism for the two LDA classifiers because selection of one LDA's output will reject the other LDA's output during the refractory

period. A simple principle of 'first come first serve' is used here for the class selection. The idea of using dwell and refractory period for asynchronous event detection was first introduced in [4] for offline event-by-event analysis of simulated asynchronous BCI, and then used in [19] for binary detection of beta oscillation. Here we use dwell and refractory period for online detection of imagery movements in real asynchronous BCI systems.

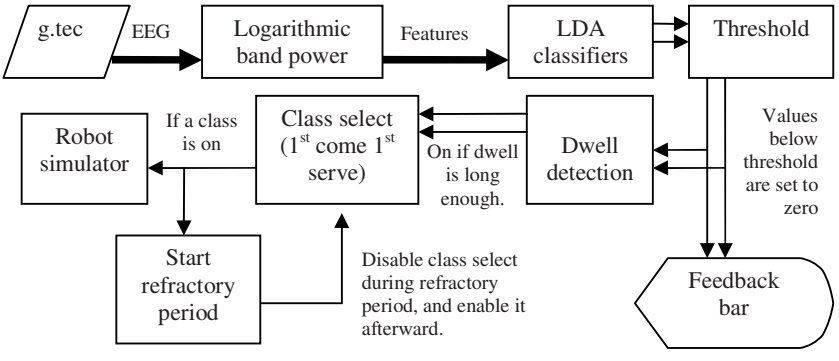


Fig. 2. An online asynchronous event detection system

It is important that the thresholds and dwell lengths are chosen for each class and each subject separately in order to optimise the performance of the whole system. This can be achieved by trial-and-error method or offline Receiver Operating Characteristic (ROC) optimisation. It is because EEG signals are non-stationary that it is also important to adapt these parameters by online training or adaptation. In the following sections an online training method is described, which adapts the threshold and dwell length values when false positive command is detected.

2.4 A Robot Simulator and Its Specifically Designed Environment

During online training, in order to detect whether there is false positive command generated by the BCI system, information about the user's control intention and timing is needed. For this purpose, a robot simulator that runs in a specifically designed environment, as shown in Fig. 3, is proposed and implemented for online asynchronous BCI experiments.

The environment is filled with hexagon grid, and robot movements are railed to the grid line. The robot simulator executes 2 commands: "turn left then move forward to the next node" or "turn right then move forward to the next node", thus the event detection system operates in 2 class mode. The task of the BCI control is to drive the simulated robot to a given target position. There is no obstacle in the environment, but the user is supposed to drive towards the target all the time. An explicit instruction is given to subjects: *At each node the user should always turn the robot to the direction that the target is located.* In the scenario given in Fig. 3, according to the above

instruction, the ideal sequence of commands to control the robot from "start" to the 1st target should be: left, right, left, right, left, right, left, right, right, left, left, right and right. It is unnecessary for subjects to plan the whole sequence in advance, as long as the subjects follow the instruction at each node. The simulator executes every detected command, including false positive ones, but the result of a false positive event will not affect future events in evaluation. This is because the correctness of each event is evaluated locally.

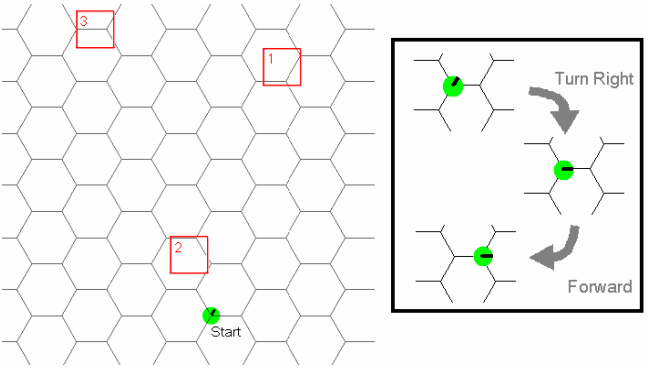


Fig. 3. (left): Hexagon grid, where the circle with a heading indicator represents the robot and the squares are targets. (right): Illustration of a "turn right then move forward to the next node" command executed.

By assuming a subject will follow the instruction, it is possible to predict what command the subject tries to send or what motor imagery movement the subject tries to perform at each node. This prediction can be used to check whether the event/command detected by the BCI is true positive or false positive. This is particularly useful for online training and performance evaluation of asynchronous BCI systems.

The key part of the user's intent prediction is the calculation of the target direction at each node. Since the robot position and target position are known. The current direction of the robot, *CD*, and the target direction, *TD,* with respect to the centre of the robot can be calculated by:

$$\begin{aligned} CD &= \text{atan}\,2(X2 - X1, Y2 - Y1)\ . \\ TD &= \text{atan}\,2(TX - X1, TY - Y1)\ . \end{aligned} \tag{1}$$

where $(X1,Y1)$ is the centre of the robot, $(X2,Y2)$ is the robot heading point, and (TX,TY) is the target location. Prediction can be obtained by comparing the values of *CD* and *TD*. However, because the output of *atan2* lies in the closed interval $[-\pi, \pi]$, simple comparison is only valid when both *CD* and *TD* are >0 or <0. To avoid this limitation, *CD* and *TD* are rotated together to the point where *CD* is aligned to the axis of $-\pi / \pi$, as shown in Fig. 4.

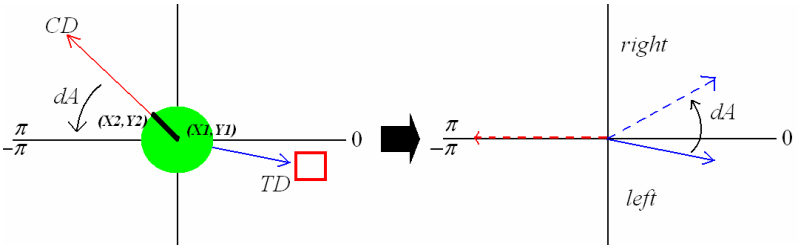


Fig. 4. Rotation of *CD* and *TD* for making a prediction

The rotation angle is as follows:

$$dA = \begin{cases} \pi - CD & CD > 0 \\ -\pi - CD & CD \le 0 \end{cases}. \tag{2}$$

After rotation the target direction is updated as follows:

$$TD = TD + dA\ . \tag{3}$$

$$TD = \begin{cases} TD - 2\pi & TD > \pi \\ TD + 2\pi & TD < -\pi \\ TD & -\pi \ge TD \le \pi \end{cases}. \tag{4}$$

Finally, the prediction of command that the user is supposed to produce at each node can be obtained based on the target direction as follows:

$$predict = \begin{cases} right & TD > 0 \\ left & TD \le 0 \end{cases}. \tag{5}$$

2.5 Online Training

To further make use of the advantage of the robot simulator and its running environment in online asynchronous BCI experiments, an adaptation scheme was implemented for online training in attempt to find the optimal parameter settings to deal with the non-stationarity of EEG signals. The general idea behind the adaptation scheme is: "If it works, do not change it", but adaptation is required when false positive is detected. We choose to adapt the dwell and threshold parameters, because they both affect the asynchronous BCI performance greatly.

During the online training period, if the detected event does not match the prediction, the following adaptation will be conducted:

- *Higher threshold and dwell are set for the triggered (false) class to prevent next false detection.*
- *On the other hand, lower threshold and dwell are set for the predicted (but not triggered) class.*

- *If the system cannot detect a command for a certain period, called maximum no control allowance, lower threshold and dwell are set for the predicted class.*

The adaptation will be controlled by learning rates. Another important issue is when to start or stop the training. Without a stopping rule, the adaptation could potentially destroy a well tuned BCI system when a minor false positive event is detected. The start and stop rules are as follows:

- *Training starts at the beginning of the online training period.*
- *Training ends when the number of consecutive true positive detections is over a preset value called #CTP.*
- *Training re-stared when the number of consecutive false positive detections is over a preset value called #CFP.*

How to choose the learning parameters will be explained in the next section in connection with experimental results.

3 Results

The system was tested online with 2 male subjects. Subject 1 had a little BCI experience before, and subject 2 was experienced in synchronous BCI experiment. Several runs of online asynchronous BCI experiments were carried out for each subject. In each run, subjects were asked to drive the robot simulator to reach 3 targets. The robot starting point and target positions were fixed and unchanged between runs in order to compare performances in a fair manner. The scenarios used in the experiments are shown in Fig. 3. However, the subjects saw only one target at a time. The current target disappeared as soon as the robot reached it, and the next target appeared at the mean time. The numbers of true positive (TP), false positive (FP), and events triggered were recorded. The performance is evaluated by accuracy defined as (TP/(TP+FP)) and the total number of events (TP+FP) happened in comparison with the minimum number of events needed (ME). The difference between the total number of events and the minimum number of events indicates the deviation of the real path from the optimal path.

3.1 Performance of the Asynchronous BCI without Online Training

The system was first tested without online training. Each subject performed 2 runs. Prior to the formal experiments, subjects took some trial runs with the present of the experiment conductor, so that the subjects understood the paradigm and requirement, and a suitable parameter setting could be found. For subject 1 (S1), 11-14 Hz and 15-30 Hz bandpass features were extracted from each channel, initial thresholds for the two LDA classifiers were set to 2.0, and initial dwell lengths were set to 1.6 seconds. For subject 2 (S2), frequency bands were chosen as 11-14 Hz and 15-25 Hz, initial thresholds were set to 1.0 for left imagery and 2.0 for right imagery, and initial dwell lengths were set to 1.4 and 1.6 seconds for left imagery and right imagery respectively. The results are shown in Tab.1 (Top).

Table 1. (Top) Performance without online training. (Bottom) Performance with online training. TP: Number of true positive event. FP: Number of false positive events. ME: Minimum number of events required to reach the target. ACC: Accuracy TP/(TP+FP). #A: Number of adaptations during online training. E/Min: Number of events per minute.

	Start to 1st target					1st to 2nd target					2nd to 3rd target				
	TP	FP	ME	ACC	E/Min	TP	FP	ME	ACC	E/Min	TP	FP	ME	ACC	E/Min
S1 - 1st run	17	6	11	73.91%	6.48	7	1	8	64.71%	8	21	12	10	63.64%	5.59
S1 - 2nd run	9	2	11	81.82%	7.02	10	3	8	76.92%	6.19	23	10	11	69.70%	6.19
S2 - 1st run	10	3	11	76.92%	7.5	9	1	8	90.00%	6.98	12	4	10	75.00%	8.65
S2 - 2nd run	11	2	11	84.62%	5.74	8	0	8	100%	7.74	15	5	10	75.00%	3.82
S1 Average=	13	4		77.87%	6.75	8.5	2		70.82%	7.10	22	11		66.67%	5.89
S2 Average=	11	2.5		80.77%	6.62	8.5	0.5		95.00%	7.36	14	4.5		75.00%	6.24

	Start to 1st target (Online Training)						1st to 2nd target					2nd to 3rd target				
	TP	FP	ME	ACC	#A	E/Min	TP	FP	ME	ACC	E/Min	TP	FP	ME	ACC	E/Min
S1 - 3rd run	10	5	11	66.67%	7	6.08	12	5	8	75.59%	7.03	12	3	11	80.00%	7.69
S1 - 4th run	9	2	11	81.82%	2	7.1	9	3	8	75.00%	7.74	10	2	10	83.33%	7.13
S2 - 3rd run	11	3	11	78.57%	3	6.51	9	1	8	90.00%	7.69	10	2	10	83.33%	4.07
S2 - 4th run	14	3	11	82.35%	3	7.56	8	1	8	88.89%	4.25	11	2	11	84.62%	4.38
S1 Average=	9.5	3.5		74.25%	4.5	6.59	11	4		75.30%	7.39	11	2.5		81.67%	7.41
S2 Average=	13	3		80.46%	3	7.04	8.5	1		89.45%	5.97	11	2		83.98%	4.23

3.2 Performance of the Asynchronous BCI with Online Training

The above experiment was repeated, except that the first part of the experiment (from start to the 1st target) was used for online training. During online training, the system adapts the thresholds and dwell lengths based on the method described in Section 2. The initial parameter settings were the same as used in 3.1. The learning rates for threshold and dwell adaptation were set to 0.25 and 0.14 seconds respectively for both subjects. Maximum no control allowance was set to 20 seconds. *#CTP* and *#CFP* were set to 5 and 3.

The results are given in Tab.1 (Bottom), which show that Subject 1 produced significantly better performance after online training. For instance, the averaged accuracy of the 3rd part of the experiment (from the 2nd target to the 3rd target) is significant improved, increased from 66.67% to 81.67%, and the number of events detected (TP+FP) is greatly reduced (close to optimal). The 2nd part of the experiment also shown improvement with averaged accuracy increased from 70.82% to 75.30% with a slightly higher number of events detected. However, first part of the experiment (start to 1st target) cannot be compared, because target position was known to the BCI system during online training.

The result from Subject 2 has also shown some improvement after online training, but not as significant as Subject 1. The reason could be that Subject 2 was very experienced in imagery arm movements, and did not make many mistakes during online training. Hence, the number of adaptations (#A) was small. Nevertheless, the averaged accuracy of the 3rd part of the experiment (from the 2nd target to the 3rd target) was increased from 75% to 83.98%.

For readers who are interested in more detailed results, some playback videos of the above online asynchronous BCI experiments are available from http://cswww.essex.ac.uk/staff/jqgan/IDEAL2007_VIDEOS/.

4 Conclusion

This paper has presented an asynchronous BCI system with online training for control of a simulated robot. Taking up the challenge in online training of asynchronous BCI systems, this paper has proposed a novel method for providing information about class labels (user's control intention and timing), which is essential for training and adapting asynchronous BCIs so as to improve the performance. This paper has also developed a method for online adaptation of the thresholds and dwell lengths of the classifiers in asynchronous BCI systems. Initial experimental results have shown the effectiveness of the proposed methods. More experiments will be conducted to further justify the methods. The current work is limited to 2 classes. The proposed experimental paradigm can be easily extended to multiple classes. Further research will be focused on online asynchronous BCI adaptation, including adaptation of all the parameters of the LDA classifiers and unsupervised adaptation during online testing when event labels are unavailable.

Acknowledgments. Chun Sing Louis Tsui is supported by Overseas Research Studentship (ORS) and University of Essex studentship. This work was partly supported by the UK EPSRC under grant EP-D030552-1. The authors would like to thank Tao Geng and Matthew Dyson for constructive discussions and help in BCI experiments.

References

1. Wolpaw, J.R., Birbaumer, N., McFarland, D.J., Pfurtscheller, G., Vaughan, T.M.: Brain-Computer Interfaces for Communication and Control. Clinical Neurophysiology. 113, 767–791 (2002)
2. Birbaumer, N., Ghanayim, N., Hinterberger, T., Iversen, I., Kotchoubey, B., Kubler, A., Perelmouter, J., Taub, E., Flor, H.: A Spelling Device for the Paralyzed. Nature 398, 297–298 (1999)
3. Pfurtscheller, G., Neuper, C.: Motor Imagery and Direct Brain-Computer Communication. Proc. IEEE 89, 1123–1134 (2001)
4. Townsend, G., Graimann, B., Pfurtscheller, G.: Continuous EEG Classification during Motor Imagery - Simulation of an Asynchronous BCI. IEEE Trans. on Neural Syst. & Rehab. Eng. 12(2), 258–265 (2004)
5. Birch, G.E., Bozorgzadeh, Z., Mason, S.G.: Initial On-Line Evaluations of the LF-ASD Brain-Computer Interface with Able-Bodied and Spinal-Cord Subjects Using Imagined Voluntary Motor Potentials. IEEE Trans. Neural Syst. & Rehab. Eng. 10(4), 219–224 (2002)
6. Millán, J.d.R., Mouriño, J.: Asynchronous BCI and Local Neural Classifiers: An Overview of the Adaptive Brain Interface Project. IEEE Trans. on Neural Syst. & Rehab. Eng. 11(2), 159–161 (2003)
7. Roberts, S., Penny, W.: Real-Time Brain Computer Interfacing: A Preliminary Study Using Bayesian Learning. Medical & Biological Engineering and Computing 38(1), 56–61 (2000)
8. Mason, S.G., Birch, G.E.: A Brain-Controlled Switch for Asynchronous Control Applications. IEEE Trans. on Biomedical Engineering 47(10), 1297–1307 (2000)

9. Krauledat, M., Dornhege, G., Blankertz, B., Curio, G., Müller, K.-R.: The Berlin Brain-Computer Interface for Rapid Response. Biomed. Tech. 49(1), 61–62 (2004)
10. Sun, S., Zhang, C.: Adaptive Feature Extraction for EEG Signal Classification. Medical and Biological Engineering and Computing 44(10), 931–935 (2006)
11. Tomioka, R., Dornhege, G., Aihara, K., Müller, K.-R.: An Iterative Algorithm for Spatio-Temporal Filter Optimization. In: 3rd International Workshop on Brain-Computer Interfaces, Graz, pp. 22–23 (2006)
12. Grosse-Wentrup, M., Gramann, K., Buss, M.: Adaptive Spatial Filters with Predefined Region of Interest for EEG Based Brain-Computer Interfaces. In: Schoelkopf, B., Platt, J.C. (eds.) Advances in Neural Information Processing Systems 19 (NIPS), MIT Press, Cambridge, MA (2007)
13. Farquhar, J.: Learning Optimal EEG Features Across Time, Frequency and Space. In: Schoelkopf, B., Platt, J.C. (eds.) Advances in Neural Information Processing Systems 19 (NIPS), MIT Press, Cambridge, MA (2007)
14. Millán, J.d.R.: On the Need for On-Line Learning in Brain-Computer Interfaces. In: IJCNN. International Joint Conference on Neural Networks, Budapest, pp. 2877–2882 (2004)
15. Buttfield, A., Ferrez, P.W., Millán, J.d.R.: Towards a Robust BCI: Error Potentials and Online Learning. IEEE Trans. on Neural Syst. & Rehab. Eng. 14(2), 164–168 (2006)
16. Sykacek, P., Roberts, S., Stokes, M.: Adaptive BCI Based on Variational Bayesian Kalman Filtering: An Empirical Evaluation. IEEE Trans. on Biomedical Engineering 51(5), 719–729 (2004)
17. Vidaurre, C., Schlögl, A., Cabeza, R., Scherer, R., Pfurtscheller, G.: Study of On-Line Adaptive Discriminant Analysis for EEG-Based Brain Computer Interfaces. IEEE Trans. on Biomedical Engineering 54(3), 550–556 (2007)
18. Blankertz, B., Müller, K.-R., Krusienski, D.J., Schalk, G., Wolpaw, J.R., Schlögl, A., Pfurtscheller, G., Millán, J.d.R., Schröder, M., Birbaumer, N.: The BCI Competition III: Validating Alternative Apporaches to Actual BCI Problems. IEEE Trans. on Neural Syst. & Rehab. Eng. 14(2), 153–159 (2006)
19. Kanoh, S., Scherer, R., Yoshinobu, T., Hoshimiya, N., Pfurtscheller, G.: Brain Switch" BCI System Based on EEG During Foot Movement Imagery. In: 3rd International Workshop on Brain-Computer Interfaces, Graz, pp. 64–65 (2006)

Fuzzy Ridge Regression with Non Symmetric Membership Functions and Quadratic Models

S. Donoso, N. Marín, and M.A. Vila

IDBIS Research Group - Dept. of Computer Science and A. I.,
E.T.S.I.I. - University of Granada, 18071, Granada, Spain
sdonosos@vtr.net, {nicm|vila}@decsai.ugr.es
http://frontdb.ugr.es

Abstract. Fuzzy regression models has been traditionally considered as a problem of linear programming. The use of quadratic programming allows to overcome the limitations of linear programming as well as to obtain highly adaptable regression approaches. However, we verify the existence of multicollinearity in fuzzy regression and we propose a model based on Ridge regression in order to address this problem.

1 Introduction

Regression analysis tries to model the relationship among one dependent variable and one or more independent variables. During the regression analysis, an estimate is computed from the available data though, in general, it is very difficult to obtain an *exact* relation.

Probabilistic regression assumes the existence of a crisp aleatory term in order to compute the relation. In contrast, fuzzy regression (first proposed by Tanaka et al. [15]) considers the use of fuzzy numbers.

The use of fuzzy numbers improves the modeling of problems where the output variable (numerical and continuous) is affected by imprecision. Even in absence of imprecision, if the amount of available data is small, we have to be cautious in the use of probabilistic regression. Fuzzy regression is also a practical alternative if our problem does not fulfill the suppositions of probabilistic regression (as, for example, that the coefficient of the regression relation must be constant).

Fuzzy regression analysis (with crisp input variables and fuzzy output variable) can be categorized in two alternative groups:

- Proposals based on the use of *possibility* concepts [10,11,12,13,16,17].
- Proposals based on the minimization of central values, mainly through the use of the least squares method[7,9].

Possibilistic regression is frequently carried out by means of the use of linear programming. Nevertheless, implemented in such a way, this method does not consider the optimization of the central tendency and usually derives a high number of crisp estimates.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 135–144, 2007.

In this work we introduce a proposal where both approaches of fuzzy regression analysis are integrated. We also show that the use of quadratic programming can improve the management of multicollinearity among input variables. To address this problem, we propose a new version of Fuzzy Ridge Regression.

The paper is organized as follows: next section presents new regression models based on the use of quadratic programming, section 3 describes a new version of Fuzzy Ridge Regression based on the methods of section 2, section 4 is devoted to presents and example, and, finally, section 5 concludes the paper.

2 Fuzzy Linear Regression

Let X be a data matrix of m variables $X_1, ..., X_m$, with n observations each one (all of them real numbers), and Y_i $(i = 1, .., n)$ be a fuzzy set characterized by a LR membership function $\mu_{Y_i}(x)$, with center y_i, left spread p_i, and right spread q_i $(Y_i = (y_i, p_i, q_i))$.

The problem of fuzzy regression is to find fuzzy coefficients $A_j = (a_j, c_{Lj}, c_{Rj})$ such that the following model holds:

$$Y_i = \sum_{j=1}^{m} A_j X_{ij} \tag{1}$$

The model formulated by Tanaka et al. [15] considers that the (fuzzy) coefficients which have to be estimated are affected by imprecision. This model intends to minimize the imprecision by the following optimization criterion [14]:

$$Min \sum_{i=1}^{n} \sum_{j=1}^{m} (c_{Li} + c_{Ri}) |X_{ij}| \tag{2}$$

subject to usual condition that, at a given level of possibility (h), the h-cut of the estimated value $\widetilde{Y_i}$ contains the h-cut of the empiric value Y_i. This restriction can be expressed by means of the following formulation[1]:

$$\sum_{j}^{m} a_j X_{ij} + (1-h) \sum_{j}^{m} c_{Rj} |X_{ij}| \geq y_i + (1-h) q_i \quad for\ i = 1, ..., n \tag{3}$$

$$\sum_{j}^{m} a_j X_{ij} - (1-h) \sum_{j}^{m} c_{Lj} |X_{ij}| \leq y_i - (1-h) p_i \quad for\ i = 1, ..., n \tag{4}$$

$$c_{Rj}, c_{Lj} \geq 0 \quad for\ j = 1, ..., m \tag{5}$$

where h is a degree of possibility for the estimate, such that

$$\mu(Y_i) \geq h \quad for\ i = 1, ..., n \tag{6}$$

The aforementioned formulation arises from the application of Zadeh's Extension Principle[18] and has been proved by Tanaka[15].

2.1 Use of Quadratic Programming

Our first approximation to the use of quadratic programming in fuzzy regression analysis is based on the interval model proposed by Tanaka and Lee[14].

If we want to minimize the extensions, taking into account that we use non symmetrical triangular membership functions, and we want to consider the minimization of the deviation with respect to the central tendency, we have the objective function

$$J = k_1 \sum_{i=1}^{n} (y_i - a^{'} X_i)^2 + k_2 (c_L X^{'} X c_L^{'} + c_R X^{'} X c_R^{'}) \tag{7}$$

where k_1 and k_2 are weights $\in [0, 1]$ that perform a very important role: they allow to give more importance to the central tendency ($k_1 > k_2$) or to the reduction of the estimate's uncertainty ($k_1 < k_2$) in the process.

The model with this objective function (7) and restrictions (3)-(5) will be called Extended Tanaka Model (ETM) in this paper and, with the parameters, $ETM(k_1, k_2)$.

Let us now focus not in the minimization of the uncertainty of the estimated results but on the quadratic deviation with respect to the empiric data. That is, we will contrast the estimated spreads with respect to the spreads of the output data (p_i and q_i).

According to this new criterion, the objective function represents the quadratic error for both the central tendency and each one of the spreads:

$$J = k_1 \textstyle\sum_{i=1}^{n} (y_i - a^{'} X_i)^2 +$$

$$+ k_2 (\sum_{i=1}^{n} (y_i - p_i - (a^{'} - c_L^{'}) X_i)^2 + \sum_{i=1}^{n} (y_i + q_i - (a^{'} + c_R^{'}) X_i)^2) \tag{8}$$

The model with objective function (8) and restrictions (3)-(5) will be called Quadratic Possibilistic Model (QPM) in this paper and, with the parameters, $QPM(k_1, k_2)$. It can be proben that this last model does not depend on the data unit.

One of the main criticisms to possibilistic regression analysis is that as the number of available data increases the length of estimated spreads also increases.

In this context, we propose a third new model, called Quadratic Non-Possibilistic (QNP), which considers the objective function (8) and which only incorporates the restriction (5).

Example 1. *We experiment with data taken from Tanaka's paper[14], where X goes from 1 to 8.*

First, we have applied the model of Kim[8] and Chang[2] with X varying from 1 to 22. The results of this analysis are depicted in Fig. 1. As can be observed, when X=15, the three curves converge ($a_i = a_i - c_{Li} = a_i + c_{Ri}$). With values higher than 15, the relationship among extreme points in the estimated membership functions reverses, so that the left extreme is higher than the right extreme (which has no sense).

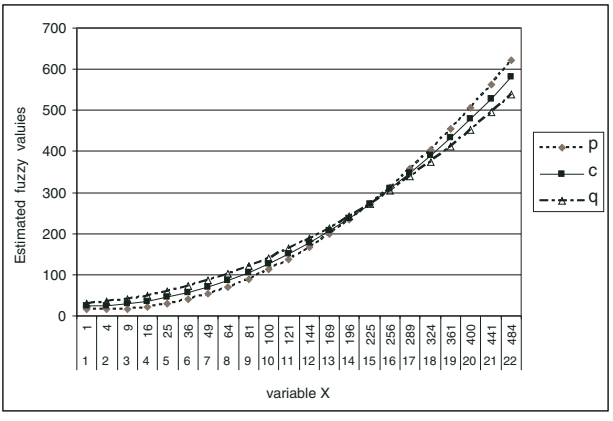


Fig. 1. Predictions with methods of Kim [8] and Chang [2]

The same experimentation with Model QNP is depicted in Figure 2. Model QNP forces the estimate's structure to be the same for both the central tendency and the fuzzy extremes. This fact, which can be seen as a restriction in the behavior of the spreads, guarantees that the inconsistencies of the previous example do not appear.

This predictive capability of the proposed model overcomes the limitation analyzed by Kim et. al [8], where the capability of prediction is restricted only to probabilistic models.

Outliers pay a determining role in the estimation of the extensions in possibilistic regression. That is the reason why we propose the use of an alternative model, the aforementioned QNP, where restrictions 3-5 are reduced to

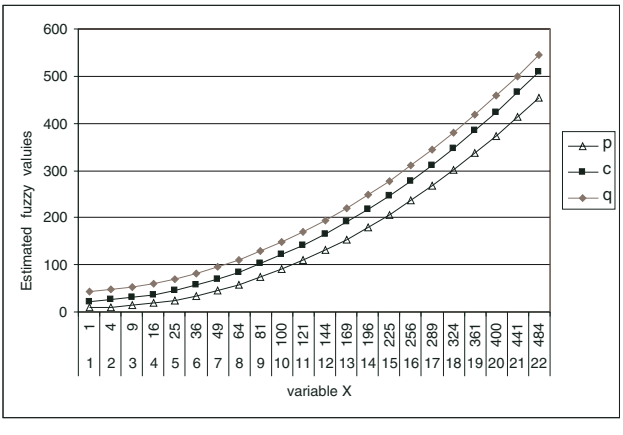


Fig. 2. Predictions with method QNP

only restriccion 5. In this new model, the estimations of extensions represent the whole set of data extensions, and not only outliers, as in the possibilistic case.

3 Fuzzy Ridge Regression

The approach based on quadratic programming analyzed in previous sections has the additional advantage of allowing the management of multicollinearity. With this approach, we can set regression methods which deal with the problem of multicollinearity among input variables, as for example, fuzzy ridge regression.

In the seminal paper of fuzzy regression, Tanaka et al. [15] stated about their concrete example "the fact that A_4 and A_5 are negative depends on the strong correlations between variables X_4 and X_5". Actually, the correlation between X_1 and X_5 is 0.95, much higher than any other value of correlation in Y and X_i, which indicates a very high multicollinearity. It can be assumed that the same distortion effect that affects probabilistic regression can be found in fuzzy regression.

The most popular probabilistic regression techniques that are usually used to deal with multicollinearity are *Principal Component Regression* and the *Ridge Regression*. Recently, papers about Fuzzy Ridge Regression has appeared in the literature which use an approach closely related to the support vector machine proposed by Vapnik [5,6].

In the area of probabilistic regression, Ridge regression can be seen as a correction of the matrix X'X. This matrix, in presence of multicollinearity, has values close to zero. It can be proven that the expected value for estimations $\widetilde{a}'\widetilde{a}$ is

$$E(\widetilde{a}'\widetilde{a}) = a'a + \sigma^2 \sum_i \left(\frac{ct}{\lambda_i}\right) \tag{9}$$

where λ_i are the eigenvalues values of X'X and ct is a constant. If these values are close to 0, the expected value for $a'a$ increases a lot, producing coefficients with high absolute value and with the opposite sign, as the comment of Tanaka et al. suggests.

The introduction of a small positive value in the diagonal of X'X moves the least value of λ_i far from zero, and, thus, the expected value for $\widetilde{a}'\widetilde{a}$ decreases.

The Ridge Regression can be seen as the addition of a new factor to the objective function. This factor depends on a parameter λ, called Rigde parameter. Ridge regression minimizes the conventional criterion of least squares in the following way [4]:

$$a^{ridge} = \min_a \left[\sum_i (y_i - \sum_j X_{ij} a_i)^2 + \lambda \sum_j a_j^2 \right] \tag{10}$$

The Ridge solutions are not equivariant under changes in the scale of the inputs.

The model of Fuzzy Ridge Regression (FRR), introduced firstly in our work [3], is formalized with the following objective function:

$$k_1 \sum_{i=1}^{n}(y_i - a^{'} X_i)^2 + k_2(\sum_{i=1}^{n}(y_i - p_i - (a^{'} - c_L^{'})X_i)^2 + \\ + \sum_{i=1}^{n}(y_i + q_i - (a^{'} + c_R^{'})X_i)^2) + \lambda(a'a) \quad (11)$$

where the *penalty* only acts on the vector of central values. This model will be called FRR (fuzzy ridge regression).

An extension of the model, including the extensions, is

$$k_1 \sum_{i=1}^{n}(y_i - a^{'} X_i)^2 + k_2(\sum_{i=1}^{n}(y_i - p_i - (a^{'} - c_L^{'})X_i)^2 + \\ + \sum_{i=1}^{n}(y_i + q_i - (a^{'} + c_R^{'})X_i)^2) + \lambda(\sum_{j=1}^{m}(k_3 a_j^2 + k_4((a_j - c_{Lj})^2 + (a_j + c_{Rj})^2)) \quad (12)$$

where $k_3 y k_4$ are constants to weight the terms, and the spreads are $a_j - c_{Lj}$ and $a_j + c_{Rj}$. This model will called $EFRR_\lambda(k_3, k_4)$ (extension of fuzzy ridge regression with parameters λ, k_3 y k_4).

There exist many proposals to choose λ. Many of them suggest varying the parameter in a certain interval, checking the behavior of the coefficients, and choosing λ when the estimates remain stable.

A more general approach can be proposed, where the λ Ridge parameter depends on each variable (λ_j with $j = 1, .., m$). In this case, the objective function is as follows

$$k_1 \sum_{i=1}^{n}(y_i - a^{'} X_i)^2 + k_2(\sum_{i=1}^{n}(y_i - p_i - (a^{'} - c_L^{'})X_i)^2 + \\ + \sum_{i=1}^{n}(y_i + q_i - (a^{'} + c_R^{'})X_i)^2) + (\sum_{j=1}^{m}\lambda_j(k_3 a_j^2 + k_4((a_j - c_{Lj})^2 + (a_j + c_{Rj})^2)) \quad (13)$$

called GFRR (generalized fuzzy regression model) with the parameters λ_j, k_3 and k_4.

3.1 Examples

Let use introduce an example of use of the previously described methods.

Example 2. *The example, similar to the one used for Tanaka [15] to illustrate the problem of multicollinearity, will be used here to experiment with the previously defined Fuzzy Ridge Regression model. We will use the method QPM, with* $k_1 = 1$ *and* $k_2 = 1$ *for our calculus.*

Figure 3 depicts the trajectory of the coefficients' centers, when the λ_i parameters are function of the diagonal of the matrix X'X (from 0 to 1 with increments of 0.1).

According to the example of Tanaka et al. (Y is the price of a house), all the coefficients must be positive (maybe with the exception of the number of Japanese rooms) because as the value of the variable increases the value of the house must also increase. The regression analysis, either of least squares or our

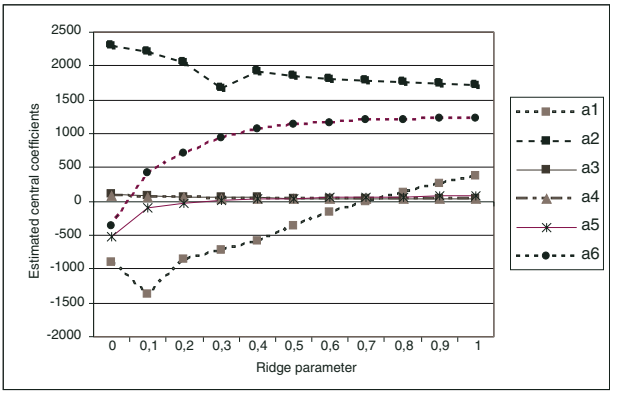


Fig. 3. Central coefficients, a_i, as λ_i increases

fuzzy regression, initially produces some negative coefficients. However, three coefficients, which initially have negative values, reach positive values.

If we suppose that the λ coefficients are constant, varying from 0 to 55, we have the trajectory for the coefficients' centers depicted in figure 4. As can be observed, one of the coefficient remains negative while the other two become positive.

In any case, the availability of more reliable coefficients permits a better knowledge of the function we are looking for, and, consequently, better conditions for the use with predictive aims.

In order to end this section, let us compare our model with the model of Hong and Hwang[5,6]. These authors do a *dual estimation* of the coefficients, with the relation:

$$\beta_{dual} = Y'(XX' + I\lambda)^{-1}X \tag{14}$$

where I is the identity matrix of range n and λ is a constant, the Ridge coefficient, whose values increase in value from 0.

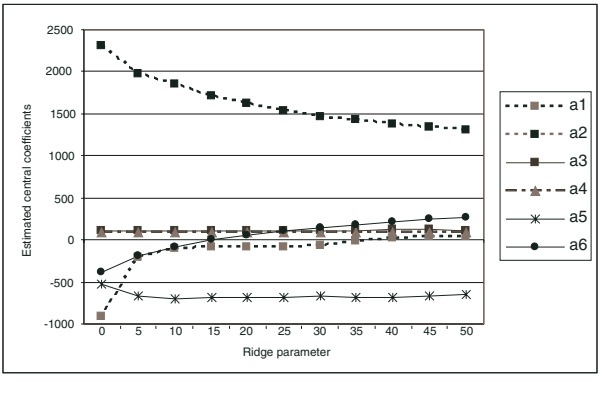


Fig. 4. Ridge central coefficients, a_i, as λ increases

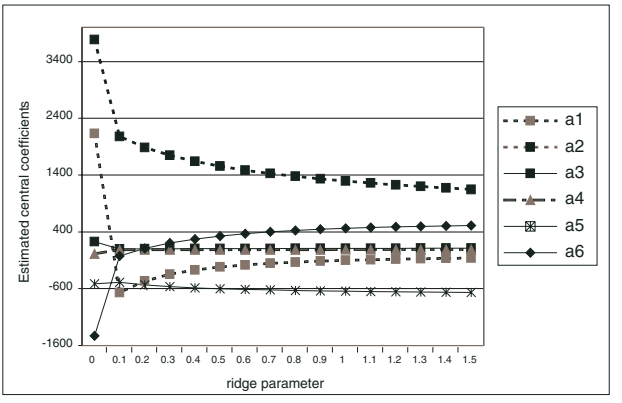


Fig. 5. *Dual Ridge Central Coefficients* of Hong and Hwang, $Beta_{dual}$ as λ increases

If we take the same data, and make λ increase from 0 to 1.5 (with increments of 0.1) we obtain the results depicted in figure 5.

These results must be contrasted with those of figure 4, where the ridge parameter is also constant. As can be observed, central coefficients have a similar behavior in both graphics: there is a positive coefficient which converges to (approximately) 1300 and a negative coefficient which converges to (approximately) -600. The other coefficients are close to zero.

However, the main difference is in the central coefficient a_1. With the method of Hong and Hwang, this coefficient has a high value when $\lambda = 0$ and is -600 when $\lambda = 0.1$. This fact does not occurs with our method.

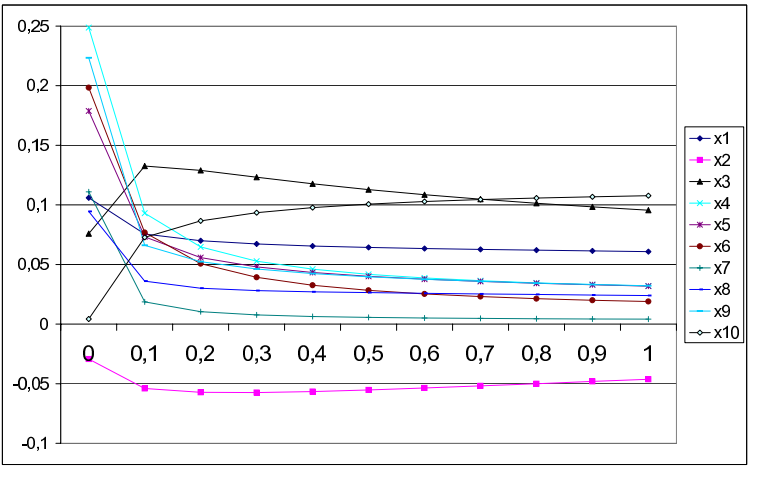


Fig. 6. *Ridge tracing, example 2*

Let us now present a second example with a higher amount of variables. We have ten demographic groups as input data and the output is the saving (positive or negative) of the whole population. For the sake of space we omit the table with numerical data.

Results with model EFRR(1,1), normalizing data according to the maximum value, have been computed with lambda varying from 0 to 1 and are shown in figure 6.

As we can see, the step from 0 to 0.1 produces the best coefficients adjustment, which, on the other hand, have a quite stable behavior. Notice that coefficients of variables x_3 and x_{10} increase their value from aprox. 0 and 0.1 to the highest values among the coefficients.

4 Conclusions

In this paper we have tried to validate the use of quadratic programming in order to obtain a good fitness in fuzzy linear regression.

To accomplish this task, we have adapted one existing model (ETM) and we have proposed two new models (QPM and QNP). Method QPM is a good choice when possibilistic restrictions are important in the problem. If we do not want to pay special attention to the possibilistic restriccions, QNP is an appropriate alternative.

We have proposed a special version of Fuzzy Ridge Regression based on our previous study on quadratic methods in order to cope with the multicollinearity problem.

References

1. Bardossy, A.: Note on fuzzy regression. Fuzzy Sets and Systems 37, 65–75 (1990)
2. Chang, Y.-H.O.: Hybrid regression analysis with reliability and uncertainty measures. Ph.D. Dissertation, University of Maryland (2001)
3. Donoso, S.: Análisis de regresión difusa: nuevos enfoques y aplicaciones. Tesis doctoral, Universidad de Granada (2006)
4. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: data mining, inference, and prediction. Springer, Heidelberg (2001)
5. Hong, D.H., Hwang, C.: Ridge regression procedure for fuzzy models using triangular fuzzy numbers. Fuzziness and Knowledge-Based Systems 12(2), 145–159 (2004)
6. Hong, D.H., Hwang, C., Ahn, C.: Ridge estimation for regression models with crisp input and gaussian fuzzy output. Fuzzy Sets and Systems 142(2), 307–319 (2004)
7. Kao, C., Chyu, C.L.: Least squares estimates in fuzzy regresion analysis. European Journal of Operation Research, 426–435 (2003)
8. Kim, B., Bishu, R.R.: Evaluation of fuzzy linear regression models by comparison membership function. Fuzzy Sets and Systems 100, 343–352 (1998)
9. Kim, K.J., Moskowitz, H., Koksalan, M.: Fuzzy versus statistical lineal regression. European Journal of Operational Research 92, 417–434 (1996)
10. Ozelkan, E.C., Duckstein, L.: Multi-objetive fuzzy regression: a general framework. Computers and Operations Research 27, 635–652 (2000)

11. Peters, G.: Fuzzy linear regression with fuzzy intervals. Fuzzy Sets and Systems 63, 45–55 (1994)
12. Redden, D.T., Woodall, W.H.: Further examination of fuzzy linear regression. Fuzzy Sets and Systems 79, 203–211 (1996)
13. Sugihara, K., Ishii, H., Tanaka, H.: Interval priorities in ahp by interval regression analysis. Europeian Journal of Operatin Research 158, 745–754 (2004)
14. Tanaka, H., Lee, H.: Interval regression analysis by quadratic programming approach. IEEE Trans. on Fuzzy Systems 6(4) (1998)
15. Tanaka, H., Uejima, S., Asai, K.: Linear regression analysis with fuzzy model. IEEE Trans. on Systems, Man, and Cybernetics 12(6), 903–907 (1982)
16. Tanaka, H., Watada, J.: Possibilistic linear systems and their application to the linear regerssion model. Fuzzy Sets and Systems 27(3), 275–289 (1998)
17. Tseng, F-M., Lin, L.: A quadratic interval logit model for forescasting bankruptcy. The International Journal of Management Science (in press)
18. Zadeh, L.A.: The concept of a linguistic variable and its application to aproxímate reasoning i, ii, iii. Information Sciences 8-9, 199–251, 301–357, 43–80 (1975)

A Subjective and Objective Integrated Method for MAGDM Problems with Multiple Types of Exact Preference Formats

Zeshui Xu and Jian Chen

Department of Management Science and Engineering
School of Economics and Management
Tsinghua University, Beijing 100084, China
Xu_zeshui@263.net

Abstract. Group decision making with preference information on alternatives has become a very active research field over the last decade. Especially, the investigation on the group decision making problems based on different preference formats has attracted great interests from researchers recently and some approaches have been developed for dealing with these problems. However, the existing approaches can only be suitable for handling the subjective preference information. In this paper, we investigate the multiple attribute group decision making (MAGDM) problems, in which the attribute values (objective information) are given as non-negative real numbers, the information about attribute weights is to be determined, and the decision makers have their subjective preferences on alternatives. The provided subjective preference information can be represented in three well-known exact preference formats: 1) utility values; 2) fuzzy preference relations; and 3) multiplicative preference relations. We first set up three constrained optimization models integrating the given objective information and each of three preference formats respectively, and then based on these three models, we establish an integrated constrained optimization model to derive the attribute weights. The obtained attribute weights contain both the subjective preference information given by all the decision makers and the objective information. Thus, a straightforward and practical method is provided for MAGDM with multiple types of exact preference formats.

1 Introduction

Decision making is a common activity in everyday life. In many real-world situations, such as economic analysis, strategic planning, medical diagnosis, and venture capital, etc. [1], multiple decision makers are usually involved in the process of decision making, and needed to provide their preference information over a finite set of feasible alternatives. Due to that each decision maker has his/her unique characteristics with regard to knowledge, skills, experience and personality, the different decision makers may express their preferences by means of different preference representation formats, such as utility values [2], fuzzy preference relation [3], multiplicative preference relation [3], etc. The issue has attracted great attention from researchers recently, and a variety of approaches have been developed to dealing with various group

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 145–154, 2007.

decision making problems with nonhomogeneous preference information. In [4], some representation models were established for group decision making problems based on the concept of fuzzy majority for the aggregation and exploitation of the information represented by means of preference orderings, utility functions, and fuzzy preference relations. For the group decision making problem, where the information about the alternatives provided by the decision makers can be presented by means of preference orderings, utility functions, and multiplicative preference relations, Herrera [5] presented a multiplicative decision model based on fuzzy majority to choose the best alternatives, taking the multiplicative preference relation as the uniform element of the preference representation. In the case where the decision makers provide their evaluations by means of numerical or linguistic assessments, Delgado et al. [6] introduced a fusion operator of numerical and linguistic information by designing two transformation methods between the numerical and linguistic domains based on the concept of characteristic values. Based on some aggregation operators (the linguistic weighted arithmetic averaging (LWAA) operator, linguistic arithmetic averaging (LAA) operator, linguistic weighted geometric averaging (LWGA) operator and linguistic geometric averaging (LGA) operator), Xu [7] presented two procedures for group decision making with multiple types of linguistic preference relations (including additive linguistic preference relations, uncertain additive linguistic preference relations, multiplicative linguistic preference relations, and uncertain multiplicative linguistic preference relations). Ma et al. [8] constructed an optimization model to integrate the four preference structures (utility values, preference orderings, multiplicative preference relations, and fuzzy preference relations) and to assess ranking values of alternatives. The prominent characteristic of the model is that it does not need to unify different structures of preferences or to aggregate individual preferences into a collective one, and it can obtain directly the ranking of alternatives.

However, the existing approaches dealing with different preferences over alternatives can only be suitable for handling the subjective preference information. That is, they can only be used to handle group decision making problems with single attribute (or criterion) and multiple alternatives, but unsuitable for the multiple attribute group decision making (MAGDM) problems which involves finding the most desirable alternative(s) from a discrete set of feasible alternatives with respect to a finite set of attributes. The MAGDM problems with preference information on alternatives generally contain both the subjective preference information given by all the decision makers and the objective information described by attribute values. In this paper, we propose a subjective and objective integrated method for the MAGDM problems with multiple types of exact preference formats in which the attribute values (objective information) are given as non-negative real numbers, and the decision makers have their subjective preferences on alternatives. The provided subjective preference information can be represented in three well-known exact preference formats: 1) utility values; 2) fuzzy preference relations; and 3) multiplicative preference relations. To do so, we organize the paper as follows. In Section 2, we present the studied MAGDM problems. Section 3 sets up three constrained optimization models integrating the given objective information and each of three preference formats respectively. Based on these three models Section 4 establishes an integrated constrained optimization model to derive the attribute weights, and then utilizes the overall attribute values to get the ranking of alternatives, and finally, Section 5 concludes the paper.

2 Problem Presentation

In this section, we describe the multiple attribute group decision making (MAGDM) problems under consideration with three exact preference formats: 1) utility values; 2) fuzzy preference relations; and 3) multiplicative preference relations. For convenience, we first let $M = \{1,2,...,m\}$, $N = \{1,2,...,n\}$, and $T = \{1,2,...,t\}$.

For a MAGDM problem, let $X = \{x_1, x_2, ..., x_n\}\ (n \geq 2)$ be a discrete set of n feasible alternatives, $D = \{d_1, d_2, ..., d_t\}$ be a finite set of decision makers, and $\lambda = (\lambda_1, \lambda_2, ..., \lambda_t)^T$ be the weight vector of decision makers, where $\lambda_k \geq 0$, $k \in T, \sum_{k=1}^{t} \lambda_k = 1$, λ_k denotes the weight of decision maker d_k (Ramanathan and Ganesh [9] proposed a simple and intuitively appealing eigenvector based method to intrinsically determine the weights for group members using their own subjective opinions). Let $G = \{G_1, G_2, ..., G_m\}$ be a finite set of attributes, $w = (w_1, w_2, ..., w_m)^T$ be the weight vector of attributes to be determined, where w_i reflects the relative importance degree of the attribute G_i, $w_i \geq 0$, $i \in M$, and $\sum_{i=1}^{m} w_i = 1$. Let $A = (a_{ij})_{m \times n}$ be the data matrix, where a_{ij} is an attribute value, which is expressed with positive real number, of the alternative $x_j \in X$ with respect to the attribute $G_i \in G$.

In general, there are benefit attributes and cost attributes in the MAGDM problems. In order to measure all attributes in dimensionless units and to facilitate inter-attribute comparisons, we introduce the following formulas to normalize each attribute value a_{ij} in data matrix $A = (a_{ij})_{m \times n}$ into a corresponding element in data matrix $R = (r_{ij})_{m \times n}$:

$$r_{ij} = a_{ij} \Big/ \sum_{j=1}^{n} a_{ij}, \text{ for benefit attribute } G_i,\ i \in M,\ \ j \in N \tag{1}$$

$$r_{ij} = (1/a_{ij}) \Big/ \sum_{j=1}^{n} (1/a_{ij}), \text{ for cost attribute } G_i,\ i \in M,\ \ j \in N \tag{2}$$

Based on the normalized data matrix R, we get the overall attribute value of the alternative $x_j \in X$ by using the additive weighted averaging operator:

$$z_j(w) = \sum_{i=1}^{m} w_i r_{ij},\ j \in N \tag{3}$$

In general, if the weight vector $w=(w_1,w_2,...,w_m)^T$ is completely known, then (3) can be used to determine the ranking of all alternatives x_j (j=1, 2, . . . , n). The greater the overall attribute value $z_j(w)$, the better the corresponding alternative x_j will be.

In addition, the decision makers also have preference information on alternatives, and the preference information provided by each decision maker is represented by one of the following exact preference formats:

1) Utility values [2]. A decision maker provides his/her preference on X as a set of m utility values, $U=\{u_1,u_2,...,u_n\}$, where $u_j\in[0,1]$ represents the utility evaluation provided by the decision maker to the alternative x_j.

2) Fuzzy preference relation [3]. A decision maker's preference information on X is described by a fuzzy preference relation $P=(p_{ij})_{n\times n}\subset X\times X$, with

$$p_{ij}\geq 0,\ \ p_{ij}+p_{ji}=1,\ \ p_{ii}=0.5,\ i,j\in N \tag{4}$$

where p_{ij} indicates the preference degree of the alternative x_i over x_j. If

$$p_{ij}=p_{ik}-p_{jk}+0.5,\ \text{for all}\ i,j,k\in N \tag{5}$$

then P is called a consistent fuzzy preference relation, which is given by [1]:

$$p_{ij}=0.5(w_i-w_j+1),\ \text{for all}\ i,j\in N \tag{6}$$

3) Multiplicative preference relation [3]. A decision maker's preference information on X is described by a multiplicative preference relation $B=(b_{ij})_{m\times m}\subset X\times X$ satisfying the following condition:

$$b_{ij}>0,\ \ b_{ij}b_{ji}=1,\ \ b_{ii}=1,\ \ i,j\in N \tag{7}$$

where b_{ij} indicates the preference degree of the alternative x_i over x_j, it is interpreted as x_i is b_{ij} times as good as x_j. If

$$b_{ij}=b_{ik}b_{kj},\ \text{for all}\ i,j,k\in N \tag{8}$$

then B is called a consistent multiplicative preference relation, which is given by

$$b_{ij}=w_i/w_j,\ \text{for all}\ i,j\in N \tag{9}$$

In the next section, we shall develop three constrained optimization models based on the objective information contained in the normalized data matrix R and each of three preference formats (utility values, fuzzy preference relations, and multiplicative preference relations) respectively.

3 Constrained Optimization Models Based on Data Matrix and Each Different Exact Preference Format

In the following, we establish the relationships between the given objective information and subjective preference information. In order to do so, we make the objective decision information (the normalized data matrix) uniform respectively with each format of the subjective decision information (utility values, fuzzy preference relations, and multiplicative preference relations).

1) Model Based on Data Matrix and Utility Values:

We first consider a special case where the information in the normalized data matrix $R=(r_{ij})_{m\times n}$ is consistent with the utility values u_j $(j=1,2,...,n)$, then all the overall attribute values $z_j(w)$ $(j=1,2,...,n)$ of the alternatives x_j $(j=1,2,...,n)$ should be equal to the utility values u_j $(j=1,2,...,n)$, respectively, that is,

$$\sum_{i=1}^{m} w_i r_{ij} = \mu_j, \text{ for all } j=1,2,...,n \tag{10}$$

which is equivalent to the following form:

$$R^T w = \mu \tag{11}$$

where $\mu=(u_1,u_2,...,u_n)^T$.

However, in general, the condition (10) (or (11)) does not hold. That is, the information in the normalized data matrix $R=(r_{ij})_{m\times n}$ is generally inconsistent with the utility values u_j $(j=1,2,...,n)$. Here, we introduce the deviation variable e_j :

$$e_j = \sum_{i=1}^{m} w_i r_{ij} - \mu_j, \text{ for all } j=1,2,...,n \tag{12}$$

Clearly, it is desirable that the overall attribute values $z_j(w)$ $(j=1,2,...,n)$ of the alternative x_j $(j=1,2,...,n)$ should be as closer to the utility values u_j $(j=1,2,...,n)$ as possible. Thus, we need to minimize the deviation variables e_j $(j=1,2,...,n)$, and then construct the following constrained optimization model:

(M-1) $J_1^* = Min \sum_{j=1}^{n} e_j^2$

$$s.t. \sum_{i=1}^{m} w_i = 1,\ w_i \geq 0,\ i=1,2,...,m$$

where $e_j^2 = (\sum_{i=1}^{m} w_i r_{ij} - \mu_j)^2$, for all $j=1,2,...,n$.

2) Model based on Data Matrix and Fuzzy Preference Relations:

In order to make the information uniform, we can utilize (6) to transform all the overall values $z_j(w)$ ($j = 1,2,..., n$) of the alternatives x_j ($j = 1,2,..., n$) into a consistent fuzzy preference relation $\overline{P} = (\overline{p}_{ij})_{n\times n}$ by using the following transformation function:

$$\overline{p}_{ij} = 0.5(z_i(w) - z_j(w) + 1), \text{ for all } i, j = 1,2,..., n \tag{13}$$

i.e.,

$$\overline{p}_{ij} = 0.5(\sum_{k=1}^{m} w_k r_{ki} - \sum_{k=1}^{m} w_k r_{kj} + 1) = 0.5(\sum_{k=1}^{m} w_k (r_{ki} - r_{kj}) + 1), \text{ for all } i, j = 1,2,..., n \tag{14}$$

If the information in the normalized data matrix $R = (r_{ij})_{m\times n}$ is consistent with the fuzzy preference relation $P = (p_{ij})_{n\times n}$, then the consistent fuzzy preference relation $\overline{P} = (\overline{p}_{ij})_{n\times n}$ should be equal to $P = (p_{ij})_{n\times n}$, i.e.,

$$0.5(\sum_{k=1}^{m} w_k (r_{ki} - r_{kj}) + 1) = p_{ij}, \text{ for all } i, j = 1,2,..., n \tag{15}$$

However, the condition (15) does not always hold. That is, the information in the normalized data matrix $R = (r_{ij})_{m\times n}$ is generally inconsistent with the fuzzy preference relation $P = (p_{ij})_{n\times n}$. Then we introduce the deviation variable e_{ij} such that

$$e_{ij} = 0.5(\sum_{k=1}^{m} w_k (r_{ki} - r_{kj}) + 1) - p_{ij}, \text{ for all } i, j = 1,2,..., n \tag{16}$$

Clearly, it is desirable that the consistent fuzzy preference relation $\overline{P} = (\overline{p}_{ij})_{n\times n}$ should be as closer to the fuzzy preference relation $P = (p_{ij})_{n\times n}$ as possible. Thus, we need to minimize the deviation variables e_{ij} $(i, j = 1,2,..., n)$, and then construct the following constrained optimization model:

(M-2) $$J_2^* = Min \sum_{i=1}^{n} \sum_{j=1}^{n} e_{ij}^2$$

$$s.t. \sum_{i=1}^{m} w_i = 1, \; w_i \geq 0, \; i = 1,2,..., m$$

where $e_{ij}^2 = [0.5(\sum_{k=1}^{m} w_k (r_{ki} - r_{kj}) + 1) - p_{ij}]^2$, for all $i, j = 1,2,..., n$.

3) Model Based on Data Matrix and Multiplicative Preference Relations:

To integrate the decision information in the normalized data matrix $R = (r_{ij})_{m\times n}$ and the multiplicative preference relation $B = (b_{ij})_{n\times n}$, we utilize (9) to transform all the overall values $z_j(w)$ ($j = 1,2,...,n$) of the alternatives x_j ($j = 1,2,...,n$) into a consistent multiplicative preference relation $\overline{B} = (\overline{b}_{ij})_{n\times n}$ by using the following transformation function:

$$\overline{b}_{ij} = z_i(w) \big/ z_j(w), \text{ for all } i, j = 1,2,..., n \tag{17}$$

i.e.,

$$\overline{b}_{ij} = \sum_{k=1}^{m} w_k r_{ki} \Big/ \sum_{k=1}^{m} w_k r_{kj}, \text{ for all } i, j = 1,2,..., n \tag{18}$$

If the information in the normalized data matrix $R = (r_{ij})_{m\times n}$ is consistent with the multiplicative preference relation $P = (p_{ij})_{n\times n}$, then the consistent multiplicative preference relation $\overline{B} = (\overline{b}_{ij})_{n\times n}$ should be equal to the multiplicative preference relation $B = (b_{ij})_{n\times n}$, i.e.,

$$\sum_{k=1}^{m} w_k r_{ki} \Big/ \sum_{k=1}^{m} w_k r_{kj} = b_{ij}, \text{ for all } i, j = 1,2,..., n \tag{19}$$

For the convenience of calculation, (19) can be transformed as:

$$\sum_{k=1}^{m} w_k r_{ki} = b_{ij} \sum_{k=1}^{m} w_k r_{kj}, \text{ for all } i, j = 1,2,..., n \tag{20}$$

However, the condition (20) does not always hold. That is, the information in the normalized data matrix $R=(r_{ij})_{m\times n}$ is generally inconsistent with the multiplicative preference relation $B = (b_{ij})_{n\times n}$. Then we introduce the deviation variable f_{ij} such that

$$f_{ij} = \sum_{k=1}^{m} w_k r_{ki} - b_{ij} \sum_{k=1}^{m} w_k r_{kj} = \sum_{k=1}^{m} w_k (r_{ki} - b_{ij} r_{kj}), \text{ for all } i, j = 1,2,..., n \tag{21}$$

Clearly, it is desirable that the consistent multiplicative preference relation $\overline{B} = (\overline{b}_{ij})_{n\times n}$ should be as closer to the multiplicative preference relation $B = (b_{ij})_{n\times n}$ as possible. Thus, we need to minimize the deviation variables f_{ij} $(i, j = 1,2,...,n)$, and then construct the following constrained optimization model:

(M-3) $J_3^* = Min \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij}^2$

$$s.t.\ \sum_{i=1}^{m} w_i = 1,\ w_i \geq 0,\ i = 1,2,...,m$$

where $f_{ij}^2 = (\sum_{k=1}^{m} w_k (r_{ki} - b_{ij} r_{kj}))^2$, for all $i, j = 1,2,...,n$.

4 Constrained Optimization Models Integrating Data matrix and All Three Different Preference Structures

Now we consider the MAGDM problem with three different exact preference structures, namely, utility values, fuzzy preference relations, and multiplicative preference relations. Without loss of generality, we suppose that:

1) The decision makers $d_k (k = 1,..., t_1)$ provide their preference information on n alternatives $x_j (j = 1,2,...,n)$ by means of the utility values $u_j^{(k)}$ $(j = 1,2,...,n;\ k = 1,..., t_1)$.

2) The decision makers $d_k (k = t_1 + 1,..., t_2)$ provide their preference information on n alternatives $x_j (j = 1,2,...,n)$ by means of the fuzzy preference relations $P^{(k)} = (p_{ij}^{(k)})_{n\times n},\ k = t_1 + 1,..., t_2$.

3) The decision makers $d_k (k = t_2 + 1,..., t)$ provide their preference information on n alternatives $x_j (j = 1,2,...,n)$ by means of the multiplicative preference relations $B^{(k)} = (b_{ij}^{(k)})_{n\times n}$, where $k = t_2 + 1,..., t$.

In the previous section, we have established three constrained optimization models based on the data matrix and each of the three different exact preference structures, namely, utility values, fuzzy preference relations, and multiplicative preference relations. Based on these three constrained optimization models, in the following we establish an integrated model to reflect both the objective decision information contained in the normalized data matrix $R = (r_{ij})_{m\times n}$ and the group opinion of all the decision makers $d_k (k = 1,..., t)$:

(M-4)
$$\overline{J}_1^* = Min \sum_{k=1}^{t_1} \sum_{j=1}^{n} \lambda_k \left(\sum_{i=1}^{m} w_i r_{ij} - \mu_j^{(k)} \right)^2$$

$$\overline{J}_2^* = Min \sum_{k=t_1+1}^{t_2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_k \left[0.5 \left(\sum_{l=1}^{m} w_l (r_{li} - r_{lj}) + 1 \right) - p_{ij}^{(k)} \right]^2$$

$$\overline{J}_3^* = Min \sum_{k=t_2+1}^{t} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_k \left[\sum_{l=1}^{m} w_l (r_{li} - b_{ij}^{(k)} r_{lj}) \right]^2$$

$$s.t.\ \sum_{i=1}^{m} w_i = 1,\ w_i \geq 0,\ \ i = 1,2,...,\ m$$

where $\lambda = (\lambda_1, \lambda_2, ..., \lambda_t)^T$ be the weight vector of the decision makers d_k $(k = 1,..., t)$, with $\lambda_k \geq 0, k \in T$, and $\sum_{k=1}^{t} \lambda_k = 1$.

By the linear equal weighted summation method [10], the model (M-4) can be transformed into the following single objective constrained optimization model:

(M-5) $\overline{J}^* = Min(\overline{J}_1 + \overline{J}_2 + \overline{J}_3)$

$$s.t.\ \sum_{i=1}^{m} w_i = 1,\ w_i \geq 0,\ \ i = 1,2,...,\ m$$

where

$$\overline{J}_1 = \sum_{k=1}^{t_1} \sum_{j=1}^{n} \lambda_k \left(\sum_{i=1}^{m} w_i r_{ij} - \mu_j^{(k)} \right)^2$$

$$\overline{J}_2 = \sum_{k=t_1+1}^{t_2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_k \left[0.5 \left(\sum_{l=1}^{m} w_l (r_{li} - r_{lj}) + 1 \right) - p_{ij} \right]^2$$

$$\overline{J}_3 = \sum_{k=t_2+1}^{t} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_k \left[\sum_{l=1}^{m} w_l (r_{li} - b_{ij} r_{lj}) \right]^2$$

which has m weight variables, a linear equality constraint, m linear inequality constraints, and a nonlinear objective function which is to be minimized.

Solving the model (M-5) by the well-known optimization software Lingo 9.0, we get the optimal objective function value $\overline{J}^*$, the optimal attribute weight vector $w^* = (w_1^*, w_2^*, ..., w_m^*)^T$. After that, by (3), we calculate the overall attribute values $z_j(w^*)$ $(j = 1,2,..., n)$, by which we can rank all the alternatives x_j $(j \in N)$ and then select the best one(s).

5 Concluding Remarks

In this paper, we have established an integrated constrained optimization model to solving the multiple attribute group decision making (MAGDM) problems with preference information on alternatives. The model integrates all the given objective information contained in the data matrix and the subjective preferences given by the decision makers over alternatives represented by three well-known exact preference

formats: 1) utility values; 2) fuzzy preference relations; and 3) multiplicative preference relations. Structurally, the model consists of a linear equality constraint and a system of linear inequality constraints, and a nonlinear objective function which is to be minimized. We can solve the model easily by using some existing optimization software packages such as the well-known optimization software Lingo 9.0. On the basis of the attribute weights derived from the established model, we have utilized the overall attribute values of alternatives to achieve the final ranking of the given alternatives so as to get the desirable decision result.

Acknowledgements

The work was supported by the National Natural Science Foundation of China (No.70571087 and No.70321001), China Postdoctoral Science Foundation (No. 20060390051), and the National Science Fund for Distinguished Young Scholars of China (No.70625005).

References

1. Xu, Z.S.: Uncertain Multiple Attribute Decision Making: Methods and Applications. Tsinghua University Press, Beijing (2004)
2. Chiclana, F., Herrera, F., Herrera-Viedma, E.: Integrating multiplicative preference relations in a multipurpose decision-making model based on fuzzy preference relations. Fuzzy Sets and Systems 122, 277–291 (2001)
3. Xu, Z.S.: A survey of preference relations. International Journal of General Systems 36, 179–203 (2007)
4. Chiclana, F., Herrera, F., Herrera-Viedma, E.: Integrating three representation models in fuzzy multipurpose decision making based on fuzzy preference relations. Fuzzy Sets and Systems 97, 33–48 (1998)
5. Herrera, F., Herrera-Viedma, E., Chiclana, F.: Multiperson decision-making based on multiplicative preference relations. European Journal of Operational Journal 129, 372–385 (2001)
6. Delgado, M., Herrera, F., Herrera-Viedma, E., Martínez, L.: Combining numerical and linguistic information in group decision making. Information Sciences 107, 177–194 (1998)
7. Xu, Z.S.: Group decision making based on multiple types of linguistic preference relations. Information Sciences 178, 452–467 (2008)
8. Ma, J., Fan, Z.P., Jiang, Y.P., Mao, J.Y.: An optimization approach to multiperson decision making based on different formats of preference information. IEEE Transactions on Systems, Man, and Cybernetics-Part A 36, 876–889 (2006)
9. Ramanahan, R., Ganesh, L.S.: Group preference aggregation methods employed in AHP: An evaluation and an intrinsic process for deriving members' weightages. European Journal of Operational Research 79, 249–265 (1994)
10. Igizio, J.P.: Goal Programming and Extensions. Health, Lexingto, MA (1976)

Energy Saving by Means of Fuzzy Systems

José R. Villar[1], Enrique de la Cal[1], and Javier Sedano[2,⋆]

[1] Computer Science Department, University of Oviedo, Gijón, Spain
villar@uniovi.es, delacal@uniovi.es
[2] Electromechanic Engineering Department, University of Burgos, Burgos, Spain
jsedano@ubu.es

Abstract. It is well known that global sustainability must begin with human actions. A reduction of the consumed energy in the heating systems is one of such possible actions. The higher the society prosperity the higher the required houses comfort, and the higher amount of energy. In Spain it is especially important as the construction rate is almost the half of that in Europe. To save energy is urgent, which means that the energy losses must be reduced.

In this paper, a multi agent system solution for the reduction of the energy consumption in heating systems of houses is presented. A control central unit (CCU) responsible of minimising the energy consumption interacts with the heaters. The CCU includes a Fuzzy Model (FM) and a Fuzzy controller (FC) and makes use of the concept of energy balance to distribute the energy between the heaters.

Results show the proposed system as a very promising solution for energy saving and comfort tracking in houses. This solution is the preliminary study to be included in a heating system product of a local company.

1 Introduction

The building rate is increasing all over the world, and this determines the resources consumption in the planet. The emerging economies are the high resources consumers, so they produce a huge impact in environment pollution [10]. For instance, the 28.4% of buildings in Europe were constructed in Spain during 2005 [1]. According to this, from 1990-2007 the trend of emissions in Spain will surpass in 20 percentual points the spanish limit in the Kyoto Agreement [2].

A direct consequence are the policies to impulse the reduction of energy consumption. In particular, the energy consumption reduction in the construction of buildings is not defined yet [4,9,17]. In buildings, heating systems represent the main energy consumption source. The thermal comfort in houses is also increasing, so more energy will be spent in heating systems.

Energy saving systems must manage a lot of variables in order to maximize the comfort while the energy needs are minimized [6,5,10,12]. Therefore, simulation

⋆ This work was funded by Spanish M. of Education, under the grant TIN2005-08386-C05.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 155–167, 2007.

tools are used for estimating the energy consumption and the comfort measures in architectural projects. Building design simulation tools can be divided in equation based models [7,10,9,11], and third part developed tools based, [4,17]. The equation based models are suitable when building materials or new construction procedures are analyzed. Simulation tools based on concentrated parameters models are more suitable when modeling the energy consumption. Some common tools based on concentrated parameters are HTB2 [18] and HOT2000 [19].

In apart, a new catalogue of dry electrical heaters will be marted by a local company in late 2007. In this work, a new Fuzzy Energy Saving Domotic System will be designed as a complement for such a product, with the aim of saving energy while keeping the comfort in the house.

This proposal includes a distributed architecture based on several distributed agents (the heaters) and a Central Control Unit (CCU). Both the CCU and the heaters will communicate using Zigbee technology. An energy saving and distributor algorithm, which makes use of a fuzzy model and a fuzzy controller, is implemented in the CCU. The energy requirements of the comfort specifications in each room is determined by means of the fuzzy model, while the fuzzy controller is used to calculate the heating power to be applied in each heater. Finally, this Energy Distribution Algorithm (EDA) determines how to share the contracted energy between the heaters in the house to achieve the desired comfort temperature level.

This work is ordered as follows. In the next section the problem is described. In Sect. 3 the Fuzzy Energy Saving Domotic System is analyzed and explained. The experiments run and the results obtained can be seen in Sect. 4. Finally, the conclusions and proposed future works are given.

2 The Problem Definition

The main goal is the design of a system for saving electrical energy for keeping the comfort level of the house considering the remaining electrical installations (but the heating installation) and the contracted electrical power. The comfort level in a house is defined as the ambient variables that the user fixes for each room: each room temperature, humidity level, etc. Houses can be constructed using different kinds of materials, different topologies, from different geographical zone, etc. Thus, here we define the standard house in Spain in Fig. 1.

Each room in the house will have at least one heater, with a given power rate. The number of heaters and the power installed in each space is defined by the spanish comfort regulations [3]. It is assumed that each heater will collaborate with the CCU in order to reduce the consumed energy while keeping the comfort in the house. The comfort specifications for each room must be given, establishing the temperature set point for every hour and day of the week. Moreover, the electrical energy contract for each house limits the amount of energy that could drain in a house. Currently, the most usual rate in Spanish houses is 4.4 KWh. Besides, the sum of the electrical power of each heater plus the power of the rest of the electrical devices in the house must not exceed the former limit. Finally,

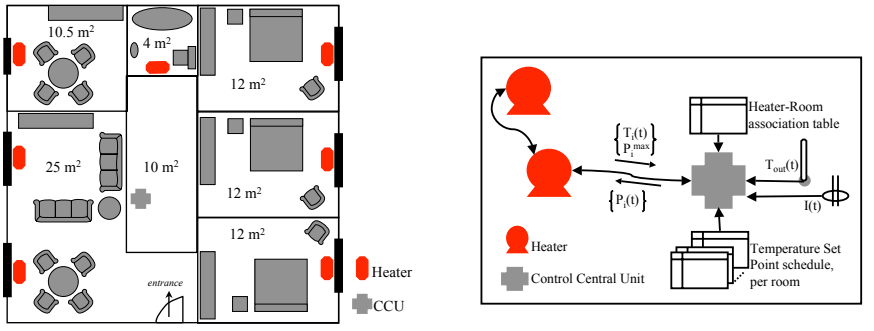


Fig. 1. The standard house designed and the architecture schema. The heaters are the gray boxes close to the windows. The CCU is in the corridor. The multiagent system schema. The CCU receives all of the information about the comfort variables from each heater. The CCU also computes and distributes the instantaneous maximum power for each heater.

the whole saving energy system must be economically viable so that consumers can afford it. Also, the system must be robust so failures in the CCU do not collapse the system.

As stated above, the model is to be included in a heating system product of a local company. The possible power rates for the heaters are fixed by fabric catalogue, and they could be 500, 1000, 1200 and 1500 Watts.

3 The Fuzzy Model for Energy Saving and Comfort

In this work, a multiagent system is proposed as a solution for the problem defined in the previous section. In few words, the heaters send to the CCU the temperature from the room, and its power rate. The CCU measures the instantaneous consumed current and the outdoor temperature. The CCU also stores the set point temperature profiles for all of the rooms in the house, and the association between rooms and heaters. Finally, the CCU computes and distributes the instantaneous maximum power for each heater. This power distribution is calculated to optimize the energy consumption in the house. In the right of Fig. 1 it is shown a schema of the system.

The power distribution between the heaters is carried out by means of an energy distribution algorithm, which makes use of a fuzzy model and a fuzzy controller to fit the maximum power for each radiator. A block diagram of the whole process is shown in Fig. 2. There are two stages in the solution: the design stage and the run stage. In the design stage, a fuzzy model (FM) and a fuzzy controller (FC) are generated. The FM has to estimate the power requirements of a generic room, and the fuzzy controller (FC) has to fit the power of a heater. To generate the learning datasets for the FM the simulation software tool HTB2 [18] has been used. The FC was designed *ad hoc*.

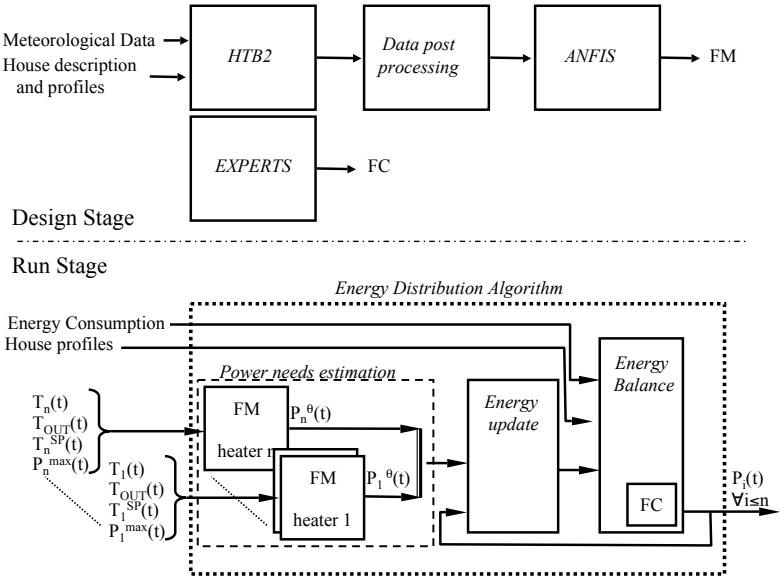


Fig. 2. The prototype black boxes diagram. In the design stage, the fuzzy model (FM) and the fuzzy controller (FC) are designed. In the run stage, the instantaneous maximum power for each heater is computed and distributed.

The instantaneous maximum power for each heater has to be determined in the run stage. The CCU receives the power rate and the temperature of the room from all of the heaters, and estimates the power requirements of each one by means of the FM. Then, the estimated energy required for heating the room and the energy used in heating the room are calculated. Finally, a balance of energy is used to infer the instantaneous heating power for each heater. The FC is used to calculate the heating power required for each heater, knowing already the room temperature error and the energy error.

The whole solution is detailed as follows. The next subsection deals with the analysis of the solution, while in Sect. 3.2 the multiagent system is roughly described. The FM and the energy distribution algorithm are detailed in Sect. 3.3 and Sect. 3.4.

3.1 The Solution Analysis

Before providing a solution for the problem, some definitions must be explained. For example, the actual definition of comfort, or the physical variables that must be measured, as well as the system architecture, or the degree of parameterization of the prototype.

Comfort is defined as the state of physical ease and absence of pain and constraint. In the problem that concerns us, comfort means the conditions of the environment of a house which helps the inhabitants to relax. There are several

methods to measure the level of comfort. For example, measuring the temperature and the humidity of a room could be used to infer the level of comfort. When talking about heating systems, the constructing regulations establish the use of the temperature in a room to measure the level of comfort. The temperature is the only data used to measure the comfort. Further studies should be carried out including other date, as the humidity, for measuring the comfort level. In apart, in each heated room there is at least one heater. As each heater includes a temperature sensor, the temperature measured in each heater will be used as the room temperature. Each heater also includes a microcontroller, and a human interface.

Besides, the outdoor temperature and the instantaneous power consumption have to be measured. The outdoor temperature should to be measure in order to reduce the energy waste as much as possible, while the power consumption should be known so the energy supplier contract limit is not exceeded.

Some hints about costs are to be considered when deciding the system architecture. First, the cost for installing the connections must be kept as low possible, not only of materials but also in manpower. Second, the cost of configuring the system should be also low: the installer must configure in a limited period of time. The configuration must be simple, with a reduced set of parameters.

The electric heaters to be installed are autonomous, that is, they are the same heaters that could be installed in a house without the energy saving system. Each heater includes a microcontroller, a temperature sensor and a human machine interface for setting it up. As stated before, a CCU would be installed to control the whole saving system. But also, the CCU would be the common interface with the system, storing the set point temperature schedule, the configuration of the house, etc.

A wireless network is used with the aim of reducing the installing costs. If the heaters are extended with such capability, then they can send to the CCU the room temperature data, and their power rate data too. Also, the outdoor temperature sensor and the instantaneous power consumption sensor could be wireless, but now both are considered wired.

The configuration costs in a wireless network of heaters are reduced to the following parameters: a network address is given to each heater, provided that it is unique in the neighbourhood of the systems; then the heater can send and receive data from the CCU.

There are some more parameters to set up in the CCU. First, assign a unique network address to the CCU. Also, name the number of rooms and their type -kitchen, living room, etc.-, and the association table between rooms and heaters. The default set point temperature schedule -in following, temperature profile- can be given by software, and the final user can easily customize. A total of 27 parameters should be given for a house as that described in Sect. 2: 8 network addresses, with 6 rooms, 6 rooms types, and 7 associations room-heaters.

The Zigbee model [15] of wireless network has been chosen as all the nodes in the network are microcontrollers, and Zigbee is a low cost microcontroller network. In this network each heater acts as a node, but also could act as a repeater

when needed, in order to get all the nodes reaching the CCU. This behaviour is dynamically determined in the network, and could not be parametrized.

3.2 The Multi Agent System

As stated before, in a Zigbee network each heater decides when to behave as a repeater. Moreover, a heater will be governed by the CCU when the network is up. But when the network is down, or when it detects that the CCU is out of service, the heater must act as a normal stand alone heater until the system recovers.

This is the reason why a mutiagent system paradigm is adopted in a similar way as proposed in [7]. In Fig. 1-right, it is shown the schema of the multiagent system, where the data flows between heaters and CCU. A heater could act as a stand alone heater, as in the network heater with or without repeater behaviour. The goal is that the house reaches the predefined comfort level.

3.3 The Estimation of the Power Requirement in a Room

The power requirements of a room is the instantaneous electric power needed to heat up the room, so it could reach the comfort level. But it is not possible to have a model for each different room. In the hypothetical case that it could be possible, there were too many parameters to be configured in the CCU that would made the installation unfeasible. That is the reason why the behaviour of the heating dynamics in a generic room is modeled. The estimation of the power requirements of a generic room has to deal with uncertain dynamics.

There are several reasons that induce the uncertainty. First, the geometric dimensions of a room are represented by the power rating of the heater or heaters installed in because this is the only known data from the room. Second, heating up a room depends on the weather, it is not the same to heat up a room in a cold winter than in a warm autumn. Finally, the occupancy profile, or the small power profile in each room introduces vagueness in the behaviour of a heating system. As a conclusion, the estimation of the power requirements of a generic room needs a model suitable to manage such vagueness.

Fuzzy logic is a well known technique for managing uncertainty [16]. By means of a fuzzy model, the power requirements of a generic room estimation is to be accomplished. The learning of the fuzzy model is carried out in the design stage. The ANFIS model [13] has been chosen to model the power requirements of a generic room, which has been proved suitable when the dataset includes data from all kind of events need to be learned, and when it is to be used in short-mid term [8,14]. The generation of the ANFIS model is shown in a block diagram in the Fig. 2. The HTB2, postprocessing and training steps are described in the following subsections.

Creating the Dataset from Realistic Simulations. As stated in [8,14], to successfully train an ANFIS model it is needed a dataset sufficiently large to include all of the situations and events be modeled. At the moment this paper

was written, the production of the heaters in the fabric is just initiating, so that it was not possible to have a physical installation as a test bed. The dataset must be generated from simulations. The HTB2 simulation software [18], in following HTB2, is a well known tool suitable to analyze the dynamics of heating systems with concentrated parameter problems as the one that concerns us [4,17].

The HTB2 is a totally parametrized simulation tool. The main output of a HTB2 simulation is the heating power requirements for each heater and room in the modeled house, but also the temperature in each room, both logged at each time interval. The materials in the construction, the volumetric measures of each room, the neighbourhood of the rooms, the orientation and geographical earth zone, the solar radiation profile, the environment data, the heating subsystems, the occupancy profile, the temperature profile for each heating subsystem, the small power devices and the light ON profiles can be fixed for each room and space. Different sample periods and the length of the simulations could be fixed too.

For generating the dataset, all of the topologies of houses and geographical zones to be covered must be defined, which are market decisions. Then, each topology and geographical zone must be defined for the HTB2, and a set of simulations for each season must be carried out using realistic profiles of occupancy, set point temperature, small power devices consume, etc.

The Postprocessing Stage. The HTB2 software generates a huge dataset. Each line of the HTB2 output dataset includes data from all of the rooms at certain time interval. This dataset must be postprocessed, so each of the examples in it contains relevant data from one room only. First, a grouping step is run, so each line in the dataset contains data from only one room. Then the grouped dataset is resampled, so only relevant examples are taken into account. Relevant examples are those that include information of dynamics. For example, when a room set point temperature is 0 there is no need to be modeled as it does not have information of the dynamics.

The outcome of the postprocessing stage is the dataset for training and testing purposes, each line in such dataset contains values of the following variables for a certain room: the temperature in the room T_i, the room set point temperature profile T^{SP}, the heating power installed in the room P_i^{max}, the power requirements for the room heater $\hat{P}_i$, the outside temperature T_{out}, the occupancy rate of the room O_i, the light power consume of the room L_i, and the small power devices consume of the room S_i. The T_i is one of the outputs of the HTB2 simulations. The values of T^{SP}, T_{out}, O_i, L_i, and S_i are the same used in the HTB2 simulation. The P_i^{max} represents all of the knowledge about a specific room, but also the maximum amount of power to be employed. The $\hat{P}_i$ and the T_i are the mayor outcome of the HTB2 simulations.

The Learning Stage. The learning phase of the ANFIS FM has been developed in Matlab [20]. The postprocessing output dataset will be used in training and validation, it is stratified in a 10-k fold cross validation schema, and the training and testing of 10 ANFIS models is carried out, provided that the dataset is large enough to contain sufficient relevant information.

As a result, 10 ANFIS models are obtained, so statistical analysis could be done. The best suite model will be chosen. The ANFIS model will estimate the power requirements of a generic room for a given environment condition. The input variables of the FM are the instantaneous inroom temperature given from a heater, the outdoor temperature, the heater power rate and the room set point temperature as stated in the comfort profile. The output of the model is the estimation of the power requirements for the room.

3.4 The Energy Distribution Algorithm

In Spain, the heating power in a house easily surpasses the 7 KW, despite the most common electric power rate contract in Spain is 4.4 KW of instantaneous consume per hour. Moreover, there are always some small power devices drawing energy: the fridge, computers, etc., so that the real instantaneous available power is lower than the contract rate. At any moment a another small power device can be in use, as example, a microwave oven.

As a conclusion, the power requirements for each room should be estimated, but also the available power should be determined, in order to distribute along the active heaters and not surpass the contracted power limit. An active heater is a heater for which its associate room has a higher temperature set point than a certain threshold. Obviously, the power distribution will be efficient if the pre defined house comfort level is reached with the minimum electric energy waste.

The solution makes use of the concept of energy balance. Energy is the capacity of the heating system to keep the comfort level. The heater *required power* is the power that the heater must spent to keep the comfort in the room. The heater *required energy* is its required power by time unit. A distribution algorithm is used, so each heater is given with a fraction of the required energy that it is allowed to spent. The *heating energy* is the fraction of the required power that a heater is allowed to spent (*heating power*) by time unit. If it is desired that the room reaches the comfort level, then the heating energy must equal the required energy for each room, that is, there must be a balance between both energies. The energy balance is carried out over a predefined period of time in order to eliminate the accumulative errors. A predefined window of 20 minutes has been adopted. This time window size would be also a requisite of the implementation in microcontrollers, where the memory is limited. For computing the energy balance over de time slicing window it is needed to store the instantaneous values of power requirements and heating power for each heater. Finally, this slicing window will reduce the impact of both lost and error values.

It is important to take into account the heaters thermal dynamics and capacity. A typical heater has a 6 minutes period from cold to full power state. From full power to cold state it takes typically 5 minutes. This means heaters do not heat with full power until the end of the dynamics. Also, it gets cold quickly because of its low thermal capacity. In the distribution algorithm those facts must be taken into account. The former could be solved with a certain correction factor. The latter implies that the active heaters must be always assigned with a minimum heating power to keep it hot. This threshold must be

determined empirically. Finally, to reduce the dynamic periods, the duty cycle must be reduced, provided it is large enough to allow them to reach the permanent state. This parameter must be determined empirically also, and it has been fixed to 3 minutes.

The adopted energy distribution algorithm is shown in the box diagram in Fig. 2, and its flow chart can be seen in Fig. 3. The first time the algorithm runs the initialization of the required energy (Er), the heating energy (Eh) and the energy error (ΔE) are carried out. Each minute the algorithm is run. In each run, the required power (Pr) for each heater are estimated by means of the FM. Then, all of the energy variables are updated. When the duty cycle runs out the heating power (Ph) for each heater is calculated. It is needed to determine the current power consumption (Pc), and to calculate the available power (Pa) as 0.85 times the contract power limit without the Pc. A FC is used for computing the heating power of each heater. The FC has been designed *ad hoc* for this problem. Finally, the Pa must be distributed between the active heaters attending to the Ph assigned to each heater, and taking into account the correction factors described before. Doing so, the Ph is updated by each three minutes, while the rest of variables Pr, Er, Eh and ΔE are updated by each minute.

The FC is a mandani fuzzy model with two inputs and one output. The temperature error in a room and the ΔE are the inputs. The temperature error is the difference between the room temperature and the set point temperature for such room. The output of the FC is the percentage of heating power to be assigned to the room heater. The variables partitions and the rules have been designed intuitively, following the ideas the experts gave. The inference method is the mean of the maximums. Future work will deals with the design of the FC by means of hybrid learning techniques.

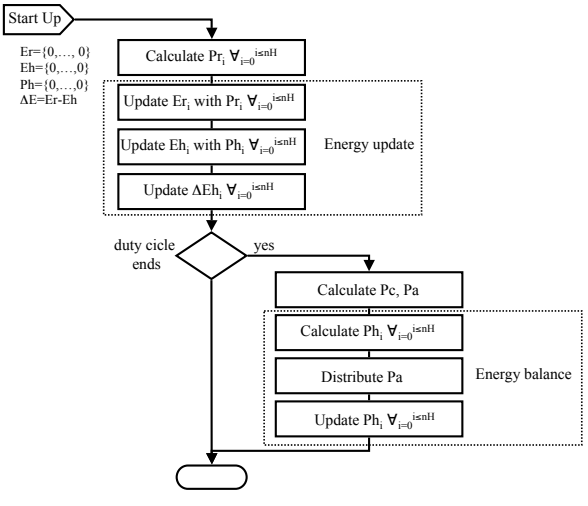


Fig. 3. The energy distributor algorithm

4 Experiments and Results

In order to test the performance of the solution, a typical Spanish construction was designed and simulated making use of the house shown in Fig. 1, which is located in the north of Spain. Future works will extend the design of the FM for each area of Spain. The weather data, such as outdoor temperature, sun hours and sun radiation are taken from statistical data in the same location. The data was gathered from a cold year in winter time.

The study was designed for a mid height house, consequently, there must be simulated houses above and under the studied house and their corresponding heat interchange. The materials and the construction type are those considered standard in an actual house building in Spain, i.e., two 6 mm thick glasses with thermal isolation and standard windows.

A suitable profile was designed for each variable in the house that is human dependent. These variables are: the temperature set point in each room, the occupancy in the house, the small power devices schedule of use, the ventilation, and the switch on lights profile. The different profiles and timetables are according to the timetable in Spain. As an example, the temperature set point profile, and the occupancy profile for a living room are shown in Fig. 4.

The simulation carried out by means of the HTB2 generated data about the power needs of each room in the house, with a sample period of 1 minute long. With the data from all of the rooms obtained from simulation, the postprocessing stage is carried out, and then the learning phase of the ANFIS FM. A 10 k-folds cross validation was used, and the validation and test errors are presented in Fig. 5.

```
!HEATSYS '1 Living Space'
  !POWER OUTPUT = 1.5
    !SPLIT = 1.0 , 0.0 , 0.0
    !CONVECTIVE CONNECTIONS
      _#1 = 1.0
    }
    !CLOCK START TIME #1 = 12:00:00 | -----ss
    !CLOCK STOP TIME #1 = 14:00:00 | -----ss
    !CLOCK START TIME #2 = 18:00:00 | mtwtf--
    !CLOCK STOP TIME #2 = 23:00:00 | mtwtf--
    !CLOCK START TIME #2 = 16:00:00 | -----ss
    !CLOCK STOP TIME #2 = 23:00:00 | -----ss
  !STAT TYPE IDEAL
  !SETPOINT HEATING= 21.0
  !TIME CONSTANT = 1200.0, 1200.0
  !DELAY LAG = 10.0
  !STAT TYPE PROPORTIONAL
  !DEADBAND = 1.0
  !BANDWIDTH = 5.0
  !STAT AIR CONNECTIONS
     _#1=1.0
  }
!END
```

```
!OCCUPANCY SPACE = 1
  !ACTIVITY LEVEL #1 = 50.0 , 0.0
  !ACTIVITY LEVEL #2 = 90.0 , 0.0
  !ACTIVITY LEVEL #3 = 150.0 , 0.0
!SPLIT = 0.80 , 0.20
!INITIAL = 0.0
!CLOCK START TIME #1 = 12:00:00 | -----ss
!CLOCK STOP TIME #1 = 14:00:00 | -----ss
  !CLOCK ACTIVITY LEVEL #1 = 2
  !CLOCK OCCUPANCY #1 = 2
!CLOCK START TIME #2 = 16:00:00 | -----ss
!CLOCK STOP TIME #2 = 18:00:00 | -----ss
  !CLOCK ACTIVITY LEVEL #2 = 2
  !CLOCK OCCUPANCY #2 = 1
!CLOCK START TIME #2 = 22:00:00
!CLOCK STOP TIME #2 = 23:30:00
  !CLOCK ACTIVITY LEVEL #2 = 2
  !CLOCK OCCUPANCY #2 = 2
!END
```

Fig. 4. HTB2 configuration file: the temperature set point -left- and the occupancy profile -rigth- for the living-room. The day of the week is specified by means of the initial letter of the day noum. A character '-' ignores the day. When no day of the week pattern is given, then the rule works for the whole week.

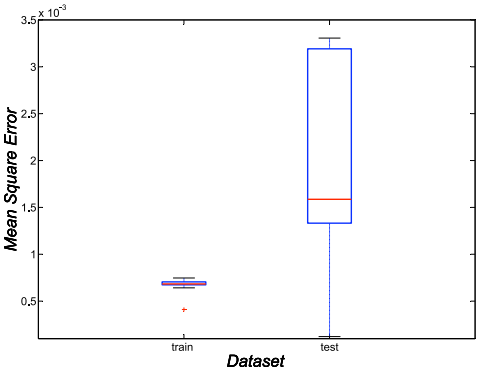
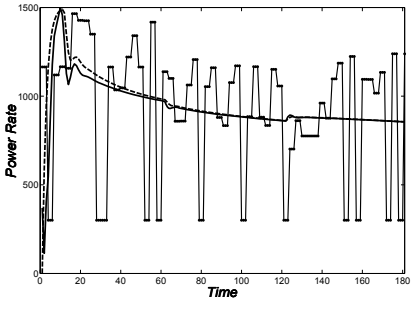


Fig. 5. The boxplot representation for the mean square error from training and testing the FM



Time Zone	Liv.	Kit.	Room1	Room2	Room3
morning	0	0.0722	0.0036	0.0196	0.0065
noon	0	0.0074	0	0	0
afternoon	0.1907	0.0853	0.0131	0.0221	0.0157

Fig. 6. In the left, the control action and the energy evolution for a room and time zone: dashed line is from HTB2, continuous line is from FM, and dotted continuous line is from EDA. In the right, the Mean Square Error of the FM for different time zones and room types.

Finally, all the system has been implemented in Matlab and simulations of the behaviour of the house have been carried out. Some facts have been taken into account, i.e., the variation in a room temperature with the power rate of a heater. In order to validate the model and algorithm above detailed the error between the temperature in each room and the corresponding HTB2 simulation room temperature are to be analyzed, but also the required energy is to be compared against the heating energy. The simulations are carried out in three different day time zones.

In Fig. 6 the control action evolution for the same room an time zone is shown. In the left side, the evolution of the instantaneous power is shown. It can be seen that the FM follows the HTB2 estimation. Finally, in Table shown in the left side of Fig. 6 the mean square error for all of the rooms and time zones can be seen.

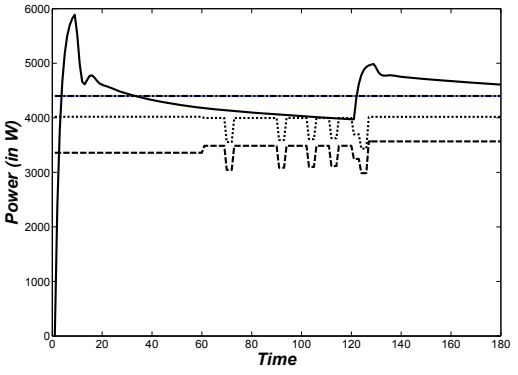


Fig. 7. The distribution and bounding action of the algorithm can be observed in the figure: bold continuous is from HTB2, dashed is from EDA, dotted is from EDA plus the small power devices profile. Straight line is the Contractual Power limit of 4400 W.

In Fig. 7 an example of energy savings can be seen. In the shown period of 240 seconds, EDA saves 378.54 kW respect contracted power energy 4.4x240 kW (a reduction of 35.85%) and 94.082 kW respect HTB2 (a reduction of 9.86%) that waste 31.847 kW above the 4.4x240 kW limit (energy losts of 3.02%).

5 Conclusions

Two mayor contributions can be extracted from the experimentation. First, it is possible to save energy by means of distributing the power between heaters. The consumed energy keeps lower than the contracted limits . On the other hand, the house ergonomics is preserved by means of the energy distribution when it is possible, that is, when the required energy is lower than the one available by the energy provider. Also, a percentage of available energy is preserved to perform contingencies.

Future work includes several ideas. The use of several geographical areas, different house topologies and data from different seasons must be analyzed. Also, the hybrid learning techniques must be employed to obtain a better FC. Different techniques, such as Fuzzy Genetic Systems, are expected to obtain a better result. Also, interchanging more information between the heaters and the CCU will allow a better required energy estimation and a better performance of the heating system. The humidity, as a comfort variable, must be studied as well. Finally, a real test bed must be implemented and the system must be validated.

References

1. Technical report, Euroconstruct (2007), http://www.euroconstruct.org/
2. Technical report, Ministerio de Medio Ambiente (2007), http://www.mma.es/

3. Boe: n. 207. Technical report, Ministerio de la Presidencia (August 29, 2007), http://www.boe.es/
4. Bojic, M., Despotovic, M., Malesevic, J., Sokovic, D.: Evaluation of the impact of internal partitions on energy conservation for residential buildings in serbia. Building and Environment (42), 1644–1653 (2007)
5. Clinch, J.P., Healy, J.D.: Valuing improvements in comfort from domestic energy-efficiency retrofits using a trade-off simulation model. Energy Economics (25), 565–583 (2003)
6. Clinch, J.P., Healy, J.D.: Cost-benefit analysis of domestic energy efficiency. Energy Policy (29), 113–124 (2001)
7. Davidsson, P., Boman, M.: Distributed monitoring and control of office buildings by embedded agents. Information Sciences 171 (171), 293–307 (2005)
8. Engin, S.N., Kuvilmaz, J., Omurlu, V.E.: Fuzzy control of an anfis model representing a nonlinear liquid-level system. Neural Computing & Applications (13), 202–210 (2004)
9. Farhanieh, B., Sattari, S.: Simulation of energy saving in iranian buildings using integrative modelling for insulation. Renewable Energy (31), 417–425 (2006)
10. Goldblatt, D.L., Hartmann, C., Dürrenberger, G.: Combining interviewing and modeling for end-user energy conservation. Energy Policy (33), 257–271 (2005)
11. Hatamipour, M.S., Mahiyar, H., Taheri, M.: Evaluation of existing cooling systems for reducing cooling power consumption. Energy and Buildings (35), 105–112 (2007)
12. Huang, C., Zou, Z., Li, M., Wang, X., Li, W., Huang, W., Yang, J., Xiao, X.: Measurements of indoor thermal environment and energy analysis in a large space building in typical seasons. Building and Environment (42), 1869–1877 (2007)
13. Jang, J.S.: Anfis: Adaptive-network-based fuzzy inference systems. IEEE Transactions on Systems, Man, and Cybernetics 23(3), 665–685 (1993)
14. Jang, J.S.: Inferring operating rules for reservoir operations using fuzzy regression and anfis. Fuzzy Sets and Systems (158), 1064–1082 (2007)
15. Kinney, P.: Zigbee technology: Wireless control that simply works. Technical report, The ZigBee Alliance (2007), http://www.zigbee.org/
16. Klir, G.J., Folger, T.A.: Fuzzy Sets, Uncertainty and Information. Prentice-Hall, Englewood Cliffs (1989)
17. Koroneos, C., Kottas, G.: Energy consumption modeling analysis and environmental impact assessment of model house in thessalonikigreece. Building and Environment (42), 122–138 (2007)
18. Lewis, P.T., Alexander, D.K.: Htb2: A flexible model for dynamic building simulation. Building and Environment (1), 7–16 (1990)
19. Natural Resources of Canada. HOT2000 User Manual (2002)
20. The Math Works. The mathworks - matlab and simulink for technical computing. Technical report, The Math Works (2007), http://www.mathworks.com/

A Comparative Study of Local Classifiers Based on Clustering Techniques and One-Layer Neural Networks*

Yuridia Gago-Pallares, Oscar Fontenla-Romero, and Amparo Alonso-Betanzos

University of A Coruña, Department of Computer Science, 15071 A Coruña, Spain
yuridiagago@hotmail.com, ofontenla@udc.es, ciamparo@udc.es

Abstract. In this article different approximations of a local classifier algorithm are described and compared. The classification algorithm is composed by two different steps. The first one consists on the clustering of the input data by means of three different techniques, specifically a k-means algorithm, a Growing Neural Gas (GNG) and a Self-Organizing Map (SOM). The groups of data obtained are the input to the second step of the classifier, that is composed of a set of one-layer neural networks which aim is to fit a local model for each cluster. The three different approaches used in the first step are compared regarding several parameters such as its dependence on the initial state, the number of nodes employed and its performance. In order to carry out the comparative study, two artificial and three real benchmark data sets were employed.

1 Introduction

Both function approximation and classification problems can be approached in one of two general ways: (a) constructing a global model that tries to fit all the input data points, or (b) dividing the input data points into several groups and learning a separate model that tries to fit in each of the local patches [3]. There are several well-known algorithms for local modelling, such as Adaptive Resonance Theory (ART) [4], Self-Organizing Maps (SOM) [5], or Radial Basis Functions (RBF) [6]. One of the first proposals of this kind of methods was the Counterpropagation model [1], that is a three-layer feed-forward network based on a Kohonen linear associator and Grossberg outstar neurons. Other representative work is the K-Winner Machine (KWM) which selects among a family of classifiers the specific configuration that minimizes the expected generalization error [2]. In training, KWM uses unsupervised Vector Quantization and subsequent calibration to label data-space partitions. In those cases in which the input data is clearly non evenly distributed, local modelling can significantly improve the overall performance. So, the best approach in these cases will consist of training a learning system for each subset of patterns that could be detected in the

* This work has been funded in part by projects PGIDT05TIC10502PR of the Xunta de Galicia and TIN2006-02402 of the Ministerio de Educación y Ciencia, Spain (partially supported by the European Union ERDF).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 168–177, 2007.

input data. However, one of the problems with local models is their recognition speed [7].

On the other hand, there are many learning methods for neural networks. One of the most popular is the backpropagation algorithm [8] for feedforward neural networks. This method however, has its drawbacks, namely possible convergence to local minima and slow learning speed. Several algorithms have been proposed in order to mitigate or eliminate these limitations [9,10]. One of this proposals for one-layer neural networks with non-linear activation functions [11] is based on linear least-squares and minimizes the mean squared error (MSE) before the output nonlinearity and a modified desired output, which is exactly the actual desired output passed through the inverse of the nonlinearity. This solution leads to the obtaining of a global optimum by solving linear systems of equations, and thus using much less computational power than with standard methods. This possibility of rapid convergence to global minimum can be taken as an important advantage for the construction of a local model once the input data is already clustered by a previous algorithm.

In this paper, a local model for classification that is composed by a clustering algorithm and a subsequent one-layer neural network of the type described above is presented. Three different clustering algorithms were tried, a k-means algorithm, a Growing Neural Gas (GNG) and a Self-Organizing Map (SOM). Their results are compared in several aspects, such as as dependence on the initial state, number of nodes employed and performance. For the comparison studies, five datasets were used. Two of them were artificially generated datasets and the other three correspond to real benchmark datasets from the UCI Learning Repository [12].

2 The Local Model

The local model designed for classification problems is composed by two different algorithms. First, a clustering method is in charge of dividing the input data set into several groups, and subsequently each of these groups is fed to a one-layer neural network of the type described in [11] with logistic transfer functions, that constructs a local model for each of them. Beside the advantages of obtaining the global minimum and a rapid speed of converge, already mentioned in the Introduction section, these neural networks conform an incremental learning. This characteristic can be of vital importance in distributed environments, such as for example in learning new signature attacks in a misuse intrusion detection system. The clustering method was implemented with three different approaches, k-means, GNG and SOM, which will be briefly described in section 2.1. Also, the construction of the local learning model has two different phases: a training phase and a testing phase, which will be described in section 2.3.

2.1 The Clustering Algorithms Implemented

As mentioned above, three different clustering algorithms were tried: k-means, GNG and SOM.

The k-means algorithm: The k-means algorithm [13] is a well-known supervised algorithm that divides the input data in k classes. It is simple to implement and works reasonably well. The main disadvantage is its dependence with the initial state, and the adjusting parameter is the number of nodes of the algorithm.

The Growing Neural Gas (GNG): The GNG is a self-organizing neural network initially proposed by Fritzke [14]. It is a very flexible structure that allows for adding/eliminating nodes of the network in execution time. The nodes of the algorithm cover all data of the input dataset, and so a modification of the original algorithm is proposed in order to adapt it for our classification purposes. The aim of the modified algorithm, published in [15], is that the nodes of the network situate in the decision region, that is, behave as the decision boundary between the classes. In this way, the modified GNG algorithm creates subregions that later could be treated by local linear classifiers. The parameters to be adjusted for this algorithm are the number of nodes, ew (ratio to adapt the nodes of the map), and lambda (node insertion frequency in the network).

The Self-Organizing Map (SOM): The Self-Organizing Map (SOM) [5] is a non-supervised and competitive learning model. The SOM defines an ordered mapping, that is a projection from a set of given input data onto a regular and usually two-dimensional grid. A data item will be mapped into the node of the map whose model is most similar to the data item, that is, has the smallest distance from the data item using a given metric, in our case, a euclidean distance. In this case, the parameters to be adjusted are the number of nodes, the number of training steps (trainlen) and four constants (a,b,c,d) that determine the vector of learning ratios and the values for the learning ratios.

2.2 The One-Layer Neural Networks

The second phase of the system is composed by a set of one-layer feedforward neural networks, one for each cluster obtained in the previous step. The goal of each network is to fit locally the data points associated to each cluster. These networks were trained using a new supervised learning method proposed in [11]. The novelty of this approach is based on the use of an alternative cost function, similar to the classical mean-squared error (MSE) function, but measuring the errors before the nonlinear activation functions and scaling them according to the corresponding operation point of the nonlinearity. The advantage of this alternative cost function is that the global optimum can be easily obtained deriving it with respect to the weights of the network and equaling these derivatives to zero. In that case, a system of linear equations can be resolved to obtain the optimal weights of the neural network. The time consumed by this method obtaining the solution is considerably lesser than that needed by the fast iterative learning methods usually employed for neural networks (quasi-Newton, conjugate gradient, etc.). This is due to the fact that method is not an iterative algorithm but rather acquires the solution in just one step.

2.3 The Construction of the Local Model

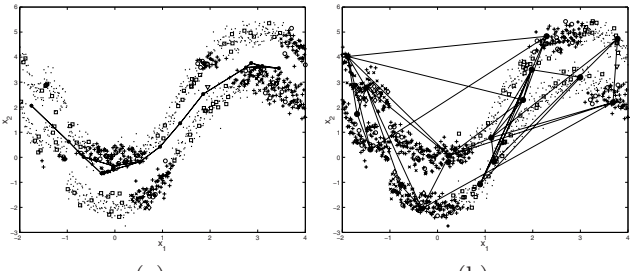
In order to obtain a final model, two different phases are needed: a training phase and a testing phase. Thus, all the datasets employed in this study were divided in training and testing sets. The training phase consists, in turn, of two stages. In the first stage, the training dataset is divided in several clusters using one of the clustering methods described in section 2.1. After this stage, several groups characterized by its centers are obtained. The second stage consists in training a set of one-layer neural networks, each one associated with one of the clusters of the previous stage. The input data for each network is the associated cluster of the first stage. The aim of this second stage is to obtain the best models for each of the clusters formed in the first stage. The testing phase consists also of two stages. In the first one, each new input data of the testing dataset is classified using the clustering method, and then, subsequently, in a second stage, that input is processed by the corresponding one-layer neural network that is determined for its group in the previous training phase.

3 Results

In order to carry out the comparative study, the three clustering algorithms were tried over five different data sets: two were artificially generated and three are benchmark data sets. The results on the tables below are obtained using a training (80% of the data set) and a test set (20% of the data set) and a further 10-fold cross-validation scheme over the training set (training set and validation set) to select the best model.

3.1 Artificial Data Sets

Data set 1: This data set contains two classes with a sinusoidal distribution plus some random noise. The set is composed by 1204 patterns, homogeneously distributed between the two classes. The input attributes correspond to the spatial coordinates of the points. Figure 1 shows the distribution of the data points in the input space with each class being represented by a different symbol.



(a) (b)

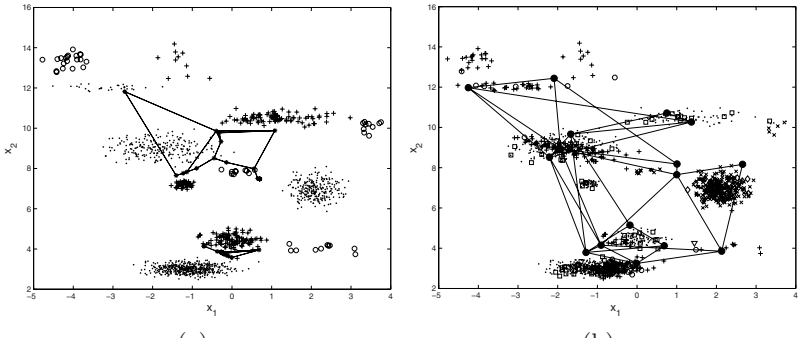
Fig. 1. Distribution of the nodes in the maps obtained for the clustering methods (a) GNG, (b) SOM, for the artificial data set 1

Table 1. Percentage of error of each method for the artificial data 1. The parameter for k-means is the number of nodes; for GNG Number of nodes/ew/lambda; and for SOM Number of nodes/trainlen/a/b/c/d.

Method	Parameters	Training (%)	Validation (%)	Test (%)
k-means	3	0.52	1.09	1.63
	8	0.04	0.20	0.29
	12	0.02	0.20	0.23
	20	0.07	0.27	0.29
	30	0.02	1.09	1.00
GNG	7/0.02/2500	1.88	2.19	3.63
	8/0.0005/5000	0.60	0.78	0.13
	20/0.0005/5000	0.22	0.47	0.50
	30/0.0005/5000	0.19	0.94	0.06
SOM	4/50/1/0.002/0.125/0.005	3.81	3.59	3.56
	9/50/1/0.002/0.125/0.005	0.22	0.44	0.50
	16/100/1/0.00004/0.000125/0.00125	0.23	0.93	0.44
	25/100/1/0.00004/0.000125/0.00125	0.12	0.16	0.69
	36/50/1/0.002/0.125/0.005	0.03	0.31	0.19

Table 1 shows the percentage of error of the three implemented clustering algorithms, for the different sets, employing different configurations of the parameters. In this case all the methods obtains similar results over the test set. Morevover, figure 1 shows graphically the distribution of the nodes in the maps obtained by the modified GNG and SOM methods. As it can be observed in this figure the behaviour of both methods is completely different. While the SOM nodes try to expand over the whole dataset, the GNG nodes situate around the decision region that separates both classes.

Data set 2: The second artificial data set is illustrated in figure 2. In this case, there are three classes and the distribution of the data set in the input space is not homogeneous. Moreover, the number of data points in each class in not balanced (class 1: 1000 patterns, class 2: 250 patterns and class 3: 50 patterns).



(a) (b)

Fig. 2. Distribution of the nodes in the maps obtained for the clustering methods (a) GNG, (b) SOM, for the artificial data set 2

Table 2. Percentage of error of each method for the artificial data 2. The parameter for k-means is the number of nodes; for GNG Number of nodes/ew/lambda; and for SOM Number of nodes/trainlen/a/b/c/d.

Method	Parameters	Training (%)	Validation (%)	Test (%)
k-means	5	0.84	0.67	1.2
	9	0.46	0.75	0.77
	15	0.45	0.75	0.80
	25	0.55	1.00	1.60
	36	0.56	1.25	1.77
GNG	5/0.0005/5000	0.40	1.08	0.53
	9/0.0005/5000	0.51	0.67	0.40
	15/0.0005/5000	0.23	0.50	0.23
	25/0.0005/5000	0.21	0.33	1.03
	36/0.0005/5000	0.18	0.67	0.80
SOM	4/100/0.00002/0.0000625/0.000625	0.84	0.92	0.63
	9/100/0.00002/0.0000625/0.000625	0.29	0.67	0.30
	16/100/0.00002/0.0000625/0.000625	0.17	0.42	0.30
	25/100/0.00002/0.0000625/0.000625	0.11	0.17	0.17
	36/100/0.00002/0.0000625/0.000625	0.09	0.25	0.23

Table 2 shows the percentage of error of the three implemented clustering algorithms. Again, all the methods present a similar performance over the test set. Furthermore, figure 2 shows graphically the distribution of the nodes in the maps obtained by the GNG and SOM methods. As in the previous data set, the modified GNG situates the nodes of the map around the decision region. Moreover, the map is divided in two different regions: one in the center of the figure and the other on the bottom.

3.2 Real Data Sets

Pima Diabetes Dataset: The first real data set is the well-known Pima Indian Diabetes database. This data set contains 768 instances and 8 attributes for each class. In this case, it is not possible to represent graphically the distribution of the data and the nodes because it is not a small dimensional space. Table 3 shows the percentage of error of the three clustering algorithms for the different sets. In this case, the best results are obtained by the k-means algorithm although the differences are not significant. Moreover, all the methods present a good behaviour in terms of generalization ability. Evidently, if many nodes or clusters are used the methods tend to overfit the training data and the generalization is affected.

Wisconsin Breast Cancer Dataset: This database was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg. It contains 699 instances, 458 benign and 241 malignant cases, but 16 were not used as they contain incomplete information. Each example is characterized by 9 attributes measured in a discrete range between 1 and 10. Table 4 shows the percentage of error of the three analyzed algorithms. For this database, the results over the test set are very similar for the best model of each type (selected using the validation set).

Table 3. Percentage of error of each method for the Pima Indian Diabetes dataset. The parameter for k-means is the number of nodes; for GNG Number of nodes/ew/lambda; and for SOM Number of nodes/trainlen/a/b/c/d.

Method	Parameters	Training (%)	Validation (%)	Test (%)
k-means	4	21.23	22.62	24.42
	8	21.21	21.82	26.36
	9	22.42	24.58	21.62
	20	21.77	24.60	21.56
GNG	4/0.00005/100	20.47	22.14	27.79
	8/0.0005/2500	20.21	26.55	20.97
	20/0.0005/2500	18.99	27.85	25.19
SOM	4/100/1/0.00004/0.000125/0.00125	20.83	24.08	22.40
	9/100/1/0.00004/0.000125/0.00125	18.02	24.57	30.65
	16/100/1/0.00004/0.000125/0.00125	16.81	27.38	27.59
	25/100/1/0.00004/0.000125/0.00125	14.37	27.51	31.17
	36/100/1/0.00004/0.000125/0.00125	12.03	29.13	29.74

Table 4. Percentage of error of each method for Breast Cancer dataset. The parameter for k-means is the number of nodes; for GNG Number of nodes/ew/lambda; and for SOM Number of nodes/trainlen/a/b/c/d.

Method	Parameters	Training (%)	Validation (%)	Test (%)
k-means	2	3.54	4.38	5.18
	5	3.54	4.62	4.26
	8	3.59	4.60	3.95
	20	4.25	5.05	6.11
GNG	2/0.0005/2500	2.80	4.41	4.30
	5/0.0005/2500	1.42	5.28	7.11
	8/0.0005/2500	1.66	7.67	5.26
	20/0.0005/2500	0.95	5.92	9.12
SOM	4/100/1/0.00004/0.000125/0.00125	2.54	4.83	4.39
	9/100/1/0.00004/0.000125/0.00125	1.22	6.14	7.37
	16/100/1/0.00004/0.000125/0.00125	0.65	9.00	6.40
	25/100/1/0.00004/0.000125/0.00125	0.07	8.56	9.47
	36/100/1/0.00004/0.000125/0.00125	0.02	10.76	11.05

Table 5. Percentage of error of each method for the Statlog dataset. The parameter for k-means is the number of nodes; for GNG Number of nodes/ew/lambda; and for SOM Number of nodes/trainlen/a/b/c/d.

Method	Parameters	Training (%)	Validation (%)	Test (%)
k-means	3	22.00	22.75	22.02
	8	21.73	22.27	23.57
	15	23.17	23.59	22.32
	20	22.35	22.94	25.01
	38	23.01	23.56	23.71
GNG	3/0.0002/1000	12.15	13.75	14.27
	7/0.00003/900	11.24	14.43	13.44
	8/0.0005/2500	10.56	13.47	14.63
	38/0.0005/1000	6.69	15.22	15.03
SOM	4/100/1/0.00004/0.000125/0.00125	13.30	15.50	13.29
	9/100/1/0.00004/0.000125/0.00125	10.49	13.30	13.03
	16/100/1/0.00004/0.000125/0.00125	8.15	13.53	12.46
	25/100/1/0.00004/0.000125/0.00125	6.10	13.58	12.74
	36/100/1/0.00004/0.000125/0.00125	4.29	12.96	15.30

StatLog Dataset: This last dataset contains images acquired by the Landsat satellite in different frequency bands (two from the visible spectra and other two from the infrared spectra). This database has 4435 patterns for the training process and 2000 for testing. Each data point is characterized by 36 attributes (the four spectras $\times$ 9 pixels from the neighbourhood). The patterns must be categorized in 6 classes. Table 5 shows the percentage of error for each method. In this data set the k-means presents a performance significantly worse than the other methods. Again as in the previous cases, the analyzed methods present a good generalization ability.

4 Comparative Results with Other Methods

Finally, in this section a comparative study with other machine learning methods is included. The results were obtained from [16]. These previous results were published using a 10-fold cross validation over the whole data set, thus the best model, selected in the previous section for each data set, was retrained using this scheme in order to carried out a comparable analysis. Table 6 shows the accuracy

Table 6. Comparative performance for PimaData, WdbcData and StatlogData

Dataset	Method	Accuracy
Pima	**k-means+1NN**	**77.87**
	Linear discriminant Analysis (LDA)	77.5-77.2
	GNG+1NN	**77.33**
	MLP+Backpropagtion	76.40
	SOM+1NN	**76.05**
	Learning vector quantization (LVQ)	75.80
	RBF	75.70
	C4.5	73.00
	Kohonen	72.70
WdbcData	**k-means+1NN**	**77.87**
	Linear discriminant Analysis (LDA)	77.5-77.2
	GNG+1NN	**77.33**
	MLP+Backpropagtion	76.40
	SOM+1NN	**76.05**
	Learning vector quantization (LVQ)	75.80
	RBF	75.70
	C4.5	73.00
	Kohonen	72.70
Statlog	Support vector machine (SVM)	97.20
	Fisher linear discriminant analysis	96.80
	MLP+Backpropagation	96.70
	Learning vector quantization (LVQ)	96.60
	k-means+1NN	**96.31**
	GNG+1NN	**96.31**
	Linear discriminant analysis (LDA)	96.00
	SOM+1NN	**95.96**
	RBF	95.90
	C4.5	93.40

(over the test set) of several methods, included the best results obtained by the approaches described in this paper, for the Pima Diabetes, Wisconsin Breast Cancer and Statlog datasets, respectively. As can be seen, the described methods obtain comparable results and in the case of the Pima Diabetes database they are among the best classifiers.

5 Conclusions

A two step local model classifier has been described in this work. The first step of the classifier is a clustering algorithm, and the second is a one-layer neural network which fits a local model for each of the clusters obtained by the first step. The one-layer neural network is of a type that obtains a global optimum by solving linear systems of equations, and thus using much less computational power than with standard methods. Three different algorithms were used for the clustering phase: k-means algorithm, GNG and SOM. The local models were tested over five different datasets. The performance obtained by the models were adequate, and in fact, are among the best results obtained when compared with other machine learning methods, as it can be seen on the previous section. In fact, for the case of the Pima Indian Diabetes dataset, our methods are the ones that obtain the best results. Regarding the three different clustering algorithm, k-means is, in average, the one that obtains worst results, probably due to its high dependence on the initial state. The other two clustering methods, GNG and SOM, are the ones obtaining best results, although their performance depends on several parameters is high, and so they are difficult to adjust. Finally, the generalization ability of all the analyzed methods is good, however it can be affected if many local models are used in the clustering phase.

References

1. Hecht-Nielsen, R.: Neurocomputing. Addison-Wesley, Reading, MA (1990)
2. Ridella, S., Rovetta, S., Zunino, R.: The k-winner machine model. In: IJCNN 2000. Int. Joint Conference on Neural Networks, vol. 1, pp. 106–111. IEEE, Los Alamitos (2000)
3. Alpaydin, E.: Introduction to Machine Learning. MIT Press, Cambridge (2004)
4. Carpenter, G., Grossberg, S.: The art of adaptive pattern recognition by a self-organizing neural network. IEEE Computer 21(3), 77–88 (1988)
5. Kohonen, T.: Self-organizing maps. Springer, Heidelberg (2001)
6. Broomhead, D.S., Lowe, D.: Multivariable functional interpolation and adaptive networks. Computer Systems 2, 321–355 (1988)
7. Bottou, L., Vapnik, V.: Local learning algorithms. Neural Computation 4, 888–900 (1992)
8. Rumelhart, D.E., Hinton, G.E., Willian, R.J.: Learning representations of back-propagation errors. Nature 323, 533–536 (1986)
9. Moller, M.F.: A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks 6, 525–533 (1993)
10. Hagan, M.T., Menhaj, M.: Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 5(6), 989–993 (1994)

11. Castillo, E., Fontenla-Romero, O., Alonso-Betanzos, A., Guijarro-Berdiñas, B.: A global optimum approach for one-layer neural networks. Neural Computation 14(6), 1429–1449 (2002)
12. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
13. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: 5-th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press, Berkeley (1967)
14. Fritzke, B.: A growing neural gas network learns topologies. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) Advances in Neural Information Processing Systems 7 (NIPS 1994), pp. 625–632. MIT Press, Cambridge, MA (1995)
15. Rodríguez-Pena, R.M., Pérez-Sánchez, B., Fontenla-Romero, O.: A novel local classification method using growing neural gas and proximal support vector machines. In: IJCNN 2007. Int. Joint Conference on Neural Networks, IEEE, Los Alamitos (2007)
16. Computational Intelligence Laboratory, Department of Informatics, Nicolaus Copernicus University: Datasets used for classification: comparison of results (Last access July 17, 2007), http://www.fizyka.umk.pl/kmk/projects/datasets.html

Filter Methods for Feature Selection – A Comparative Study*

Noelia Sánchez-Maroño, Amparo Alonso-Betanzos,
and María Tombilla-Sanromán

University of A Coruña, Department of Computer Science, 15071 A Coruña, Spain
nsanchez@udc.es, ciamparo@udc.es, infmts00@ucv.udc.es

Abstract. Adequate selection of features may improve accuracy and efficiency of classifier methods. There are two main approaches for feature selection: wrapper methods, in which the features are selected using the classifier, and filter methods, in which the selection of features is independent of the classifier used. Although the wrapper approach may obtain better performances, it requires greater computational resources. For this reason, lately a new paradigm, hybrid approach, that combines both filter and wrapper methods has emerged. One of its problems is to select the filter method that gives the best relevance index for each case, and this is not an easy to solve question. Different approaches to relevance evaluation lead to a large number of indices for ranking and selection. In this paper, several filter methods are applied over artificial data sets with different number of relevant features, level of noise in the output, interaction between features and increasing number of samples. The results obtained for the four filters studied (ReliefF, Correlation-based Feature Selection, Fast Correlated Based Filter and INTERACT) are compared and discussed. The final aim of this study is to select a filter to construct a hybrid method for feature selection.

1 Introduction

Enhancement of generalization, i.e., the performance of the learning algorithm over the test set, often motivates feature selection, which consists on detecting the relevant features and discarding the irrelevant features. Feature selection has several advantages [1], such as:

- improving the performance of the machine learning algorithm.
- data understanding, gaining knowledge about the process and perhaps helping to visualize it.
- data reduction, limiting storage requirements and perhaps helping in reducing costs.
- simplicity, possibility of using simpler models and gaining speed.

* This work has been funded in part by Project PGIDT05TIC10502PR of the Xunta de Galicia and TIN2006-02402 of the Ministerio de Educación y Ciencia, Spain (partially supported by the European Union ERDF).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 178–187, 2007.

There are two main models that deal with feature selection: filter methods and wrapper methods [2]. While wrapper models involve optimizing a predictor as part of the selection process, filter models rely on the general characteristics of the training data to select features with independence of any predictor. Wrapper models tend to give better results but filter methods are usually computationally less expensive than wrappers. So, in those cases in which the number of features is very large, filter methods are indispensable to obtain a reduced set of features that then can be treated by other more expensive feature selection methods. In fact, this is the basis of the most recent hybrid algorithms for feature selection, that try to take the strong points of both previous approaches [1,3]. One of the problems that is needed to be faced is which filter gives the best relevance index for each case, and this is not an easy to solve question. Different approaches to relevance evaluation lead to a large number of indices for ranking and selection [1]. In this paper, several filter methods are applied over several synthetic problems to test its effectiveness under different situations: increasing number of relevant features and samples, noisy output and interaction between features.

2 The Filter Methods Used

As described above, filter methods carry out the feature selection process as a pre-processing step with no induction algorithm. The general characteristics of the training data are used to select features (for example, distances between classes or statistical dependencies). This model is faster than the wrapper approach and results in a better generalization because it acts independently of the induction algorithm. However, it tends to select subsets with a high number of features (even all the features) and so a threshold is required to choose a subset.

In this paper, a comparison over several artificial problems is carried out. The filter methods selected are the following:

2.1 Relief

The original RELIEF algorithm [4] estimates the quality of attributes according to how well their values distinguish between instances that are near to each other. For this purpose, given a randomly selected instance, $\mathbf{x}_i = \{x_{1i}, x_{2i}, , x_{ni}\}$, RELIEF searches for its two nearest neighbours: one from the same class, called nearest hit H, and the other from a different class, called nearest miss M. It then updates the quality estimate for all the features, depending on the values for $\mathbf{x}_i$, M, and H. The original RELIEF can deal with discrete and continuous features but is limited to two-class problems. An extension, ReliefF [5], not only deals with multiclass problems but is also more robust and capable of dealing with incomplete and noisy data. ReliefF was subsequently adapted for continuous class (regression) problems, resulting in the RReliefF algorithm [6]. The Relief family of methods are specially attractive because they may be applied in all situations, have low bias, include interaction among features and may capture local dependencies that other methods miss.

2.2 Correlation-Based Feature Selection, CFS

Correlation based Feature Selection (CFS) is a simple filter algorithm that ranks feature subsets according to a correlation based heuristic evaluation function [7]. The bias of the evaluation function is toward subsets that contain features that are highly correlated with the class and uncorrelated with each other. Irrelevant features should be ignored because they will have low correlation with the class. Redundant features should be screened out as they will be highly correlated with one or more of the remaining features. The acceptance of a feature will depend on the extent to which it predicts classes in areas of the instance space not already predicted by other features. CFS's feature subset evaluation function is:

$$M_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}},$$

where M_S is the heuristic "merit" of a feature subset S containing k features, $\overline{r_{cf}}$ is the mean feature-class correlation ($f \in S$) and $\overline{r_{ff}}$ is the average feature-feature intercorrelation. The numerator of this equation can be thought of as providing an indication of how predictive of the class a set of features is; and the denominator of how much redundancy there is among the features.

2.3 Fast Correlated Based Filter, FCBF

The fast correlated-based filter (FCBF) method [8] is based on symmetrical uncertainty (SU) [9], which is defined as the ratio between the information gain (IG) and the entropy (H) of two features, x and y:

$$SU(x, y) = 2\frac{IG(x/y)}{H(x) + H(y)}, \tag{1}$$

where the information gain is defined as:

$$IG(x/y) = H(y) + H(x) - H(x, y),$$

being $H(x)$ and $H(x, y)$ the entropy and joint entropy, respectively. This method was designed for high-dimensionality data and has been shown to be effective in removing both irrelevant and redundant features. However, it fails to take into consideration the interaction between features.

2.4 INTERACT

The INTERACT algorithm [10] uses the same goodness measure as FCBF filter, i.e., SU in (1), but it also includes the consistency contribution (c-contribution). C-contribution of a feature is an indicator about how significantly the elimination of that feature will affect consistency. The algorithm consists of two major parts. In the first part, the features are ranked in descending order based on their SU values. In the second part, features are evaluated one by one starting from the end of the ranked feature list. If c-contribution of a feature is less than an established threshold, the feature is removed, otherwise it is selected. The authors stated in [10] that INTERACT can thus handle feature interaction, and efficiently selects relevant features.

3 Synthetic Datasets

In order to determine the effectiveness of each filter at different situations a synthetic dataset was generated. The idea of this synthetic problem was suggested by the work in [11]. Consider m pairs of training samples:

$$\{\mathbf{x}_1, y_1\}, \{\mathbf{x}_2, y_2\}, \ldots, \{\mathbf{x}_m, y_m\},$$

where $\mathbf{x}_i = [x_{1i}, x_{2i}, \ldots, x_{ni}]^T$ is a $n-$dimensional feature vector representing the i^{th} training sample, and $y_i \in [-1, 1]$ is the class label of $\mathbf{x}_i$. The number of samples considered was 400. In the original problem [11], the features were uniformly distributed in the rank $(0, 1)$, however most of the filter methods required discrete features in order to work properly. Although discretization methods, such Fayyad and Irani's MDL method [12], can be applied as a preprocessing step to overcome this problem, it was observed that this process discards some features (for example, by assigning a unique value to them) and therefore different filter methods can lead to the same set of features. Then, and in order to avoid that, a set of 200 discrete features uniformly distributed in the rank $[0, 5]$ was used.

Different experiments were carried out based on that set of input features to test the behavior of the filters. Four different situations were considered: a)different number of relevant features, b) adding noise to the output, c) interaction between features and d) different ratios between the number of samples and the number of features.

To tackle with the first two situations, the desired output was estimated as:

$$\hat{y}_i = \sum_{j=1}^{R} a_j x_{ji}, \tag{2}$$

where R is the number of relevant features considered from the whole set and a_j is randomly fixed to -1 or 1. In order to achieve a perfect balanced binary classification problem, the desired output $\mathbf{y}$ was evaluated as:

$$y_i = \begin{cases} \text{-1} & \text{if } y_i < \text{median}(\hat{\mathbf{y}}) \ , \\ 1 & \text{if } y_i >= \text{median}(\hat{\mathbf{y}}) \ . \end{cases} \tag{3}$$

The number of relevant features can be 10, 20, 30 or 40, and the percentage of noise varies from 0% to 20%. Note that, as a binary classification problem, adding noise to the output means assigning some outputs to the wrong class.

The equation in (2) does not consider interaction between variables. Therefore, the following variations were taken into account:

$$y_i = \sum_{j=1}^{R} a_j x_{ji} + b_1(x_{f_1} x_{f_2}) + b_2(x_{f_2} x_{f_3}), \tag{4}$$

$$y_i = \sum_{j=1}^{R} a_j x_{ji} + b_1(x_{f_1} x_{f_2} x_{f_3}) \tag{5}$$

In these cases, R changes from 10 to 20 and the desired output is without noise. Different problems were proposed based on equations (4) and (5), considering that the features x_f can be in both the set of R relevant features and in the set of irrelevant features. For example, one of the considered problems was:

$$y_i = \sum_{j=1}^{10} a_j x_{ji} + b_1(x_3 x_4) + b_2(x_4 x_6),$$

that involves the features $\{x_3, x_4, x_6\}$, and all of them are relevant by its own because they are included in the first term of the equation (4). The goal of this kind of problems was to test if the relevance of those features varies. On the other hand, the following problem adds new relevant features:

$$y_i = \sum_{j=1}^{10} a_j x_{ji} + b_1(x_{13} x_{14}) + b_2(x_{14} x_{15}).$$

4 Experimental Results

The filters used for this study, except INTERACT, are available at the data mining software WEKA [13]. INTERACT can be downloaded from [14]. Tables 1, 2

Table 1. Results for ReliefF. R indicates the number of relevant features and N is the percentage of noisy outputs.

R	N (%)	10-first selected features	% of success
10	0	**4**, **7**, 71, 39, 173, **1**, **3**, 176, **6**, 190, 115 ...	50
	5	**7**, **3**, **4**, **8**, 71, 109, 139, 190, 161, 176...	40
	10	**8**, 173, 121, 169, 71, 137, **7**, 148, 161, 176...	20
	15	148, 72, 49, 11, 118, 135, 158, 27, 38, **10**...	10
	20	148, 79, 76, 133, 158, **1**, 44, 112, 34, 72...	10
20	0	**10**, **6**, 199, 112, **17**, **18**, **9**, 39, **19**, 91 ...	40
	5	**18**, 112, 199, **10**, **1**, 155, 59, **17**, **14**, 102...	30
	10	**1**, **18**, 112, **2**, 115, 40, 160, 63, 184, 48...	25
	15	68, 111, 176, **5**, 39, 80, 131, 136, 199, 63...	20
	20	179, 49, 92, 150, 177, 119, 182, 50, 83, 167...	5
30	0	**1**, 188, 19, 182, 155, 17, 191, 31, 42, 104...	23, 3
	5	**17**, **1**, 73, 191, 93, 184, 32, 19, 182, 31...	23, 3
	10	191, **17**, 182, 104, **1**, 174, 138, 144, 75, 107...	20
	15	**13**, 75, 177, 71, 44, **19**, **8**, **20**, 100, 93...	16.6
	20	91,146,100,162,31,84,184,93,168,150...	16, 6
40	0	**32**, 44, **17**, 107, 90, 128, **33**, **35**, 74, **10**...	25
	5	**32**, 74, 125, 85, 44, **35**, 43, 90, 101, **17**...	25
	10	**32**, 68, 125, **17**, 44, **29**, **35**, **25**, 43, **8**...	27
	15	45, 152, 44, 99, 111, **32**, 123, 61, 134, **17**...	22, 5
	20	41, 45, **40**, 83, 93, 134, **28**, 59, 49, 70...	12, 5

and 3 show the results achieved when different number of relevant features and degrees of noise are taken under consideration with ReliefF, FCBF and CFS, respectively. INTERACT was not considered because of its similarity to FCBF. In those tables, R indicates the number of relevant features and N the percentage of noise considered. The features are consecutively labeled from 1 to 200 and the relevant features are placed at the beginning. So, if R is equal to 10, the relevant features are $\{x_1, x_2, \ldots, x_{10}\}$ and all the others are irrelevant. The percentage of success shown in the last column of Tables 1-2 indicates the percentage of correctly selected features between the R relevant ones; for example, in the first row of Table 1 the features 1 to 10 should be detected as relevant, however only 5 of them are included $\{4, 7, 1, 3, 6\}$.

As it is stated in the literature, ReliefF and FCBF return the whole set of features ranked according to its "internal" measures and they require thresholds to select a subset. Tables 1-2 show the first positions of the ranked lists. It is important to remark the variability observed in the higher values of the ranked list from one problem to another. Using ReliefF, the feature placed at the first position gets a value equal to $0,0375$ when solving the simplest problem, R=10-N=0, however this first value changes to $0,0278$ in the most difficult case, R=40-N=20. The differences are even more drastic using FCBF, being the first values

Table 2. Results for FCBF. R indicates the number of relevant features and N is the percentage of noisy outputs.

R	N (%)	10-first selected features	%success
10	0	**8**, **3**, **4**, **7**, **2**, **10**, **6**, **1**, **9**, **5** ...	100
	5	**2**, **4**, **3**, **8**, **7**, **10**, **1**, **6**, **9**, 25...	90
	10	**2**, **8**, 94, **3**, **6**, **1**, 38, 71, 145, 7 ...	60
	15	145, 117, 72,11, **8**, **10**, **2**, **6**, 168, 94...	40
	20	44, 71, 72, 145, **2**, 93, 175, 22, 26, 47...	10
20	0	**8**, **17**, **14**, **11**, **6**, **20**, 191, **4**, 39, 172...	75
	5	**8**, **17**, **20**, **14**, 172, **16**, **11**, **18**, 38, **3**...	70
	10	**17**, 38, **11**, **8**, **1**, **18**, 172, 104, **3**, **15**...	65
	15	**17**, **1**, 39, **2**, 115, 68, **10**, **16**, 93, 29...	40
	20	**8**, **16**, **17**, 127, 120, 130, **1**, 35, 200, 72...	25
30	0	**26**, **14**, **8**, **15**, **30**, **11**, **28**, **17**, **1**, **19**...	60
	5	**14**, **17**, **30**, **11**, **26**, **8**, **15**, **20**, 119, **21**...	50
	10	147,**8**, **17**, **26**, 38, 125, 104, **3**, **30**, 138...	43
	15	145, **21**, **8**, 189, **18**, 83, **17**, **23**, 105, **3**...	30
	20	145, **17**, 93, 53, **21**, **9**, 191, 71, 150, 195...	17
40	0	**26**, 34, 35, **17**, **14**, **19**, 33, 182, **30**, 134...	52
	5	**26**, **35**, **11**, 115, **19**, **28**, **14**, **34**, **30**, **33**...	47
	10	**35**, **28**, 182, 149, **30**, 1432, 102, **17**, **19**...	40
	15	**35**, 149, 144, **32**, 69, 153, 167, 182, **11**, **20**...	35
	20	172,193,83,64,169,125,93,87,150,69...	25

Table 3. Results for CFS filter. S is the subset of features that returns, $|S|$ is its cardinality and %R stands for the percentage of relevant features included in S.

R	N (%)	Relevant Features ∈ S	$\|S\|$	%R
10	0	[1-10]	14	100
	5	[1-10]	12	100
	10	[1-10]	35	100
	15	1,2,3,6,8,10	35	60
	20	2,8	37	20
20	0	[3-11],[14-20]	22	80
	5	1,[3-8],10,11,[11-20]	26	80
	10	1,2,3,[5-11],[14-18],20	25	80
	15	[1-5],[8-11],[13-17],19,20	39	80
	20	1, 2, 8, 13, 16, 17, 20	56	35
30	0	1, 4, 8, 9, 11, 13, 14, 15, 17, 20, 21, 23, 25, 26,28,30	22	50
	5	1, 3, 8, 11, 13, 14, 15, 17, 20, 21, 23, 26, 28, 30	27	47
	10	1, 3, 8, 14, 15, 16, 17, 18, 20, 21, 23, 26, 30	30	43,3
	15	3, 4, 8, 10, 13, 17, 18, 21, 23, 26, 27	36	36,3
	20	9, 17, 21, 25, 27	36	16,6
40	0	6, 8, 10, 11, 13, 14, 15, 17, 19, 20, 21, 23, 25, 26, 28, 30, [32- 35], 37	37	52,5
	5	4, 8, 10, 11, 14, 15, 17, 19, 20, 21, 23, 25, 26, 28, 29, 30, [32-35], 37	43	52,5
	10	10, 11, 14, 17, 19, 21, 23, 25, 26, 28, 30, 31, 32, 33, 35, 36	32	40
	15	5, 10, 11, 14, 17, 20, 21, 24, 26, 28, 29, 30, 31, 32, 35, 38	46	40
	20	1 , 11, 12, 13, 18, 27, 28, 29, 32, 35	43	25

$0,0375$ and $0,0169$, respectively. This fact remarks the difficulty of finding a proper threshold for each problem.

On the other hand, CFS returns a subset of features. In Table 3 can be observed that the features selected are the correct ones for the simplest cases, R=10, N=0, 5 and 10. However, the size of the selected subset increases considerably when the level of noise increases. Note that the last column of Table 3 shows the percentage of relevant features included in the subset S returned by the filter method. This performance measure is different from the one used for ReliefF and FCBF and therefore a fair comparison is not directly possible. However, it can be stated that both FCBF and CFS exhibit good performance results in the examples considered.

Tables 4 and 5 show the results obtained with interaction problems (see equations 4-5). The first columns of both tables present the different interaction problems considered. Similarly to a_j in (2), the values of b_1 and b_2 in equations (4)-(5) were equal to $0,2$ or $-0,2$, randomly chosen, unless for test number 7 in which their value was $0,4$. INTERACT was selected instead of FCBF because it is better suited for this type of problems [10]. Table 4 presents the complete set of features obtained for INTERACT and CFS. It can be noticed that, in test 2, both methods are able to detect the added features $\{x_{13}, x_{14}, x_{15}\}$. Besides, CFS seems to detect

Table 4. Results obtained with Interact and CFS using different problems of interaction between features based on equations (4) and (5). f_1, f_2 and f_3 indicate the indexes of the selected features. Tests shown over the double line use 10 relevant features, while those below it use 20 relevant features.

Test	Eq.	f_1	f_2	f_3	INTERACT	CFS
1	(5)	1	7	24	2,3,4,5,10,**24**	[1-5],8,9,10,12,**24**,35,180
2	(5)	13	14	15	$4, 6, 8, \mathbf{13}, \mathbf{14}, \mathbf{15}$	$1, 2, 4, [6-9], \mathbf{13}, \mathbf{14}, \mathbf{15}, 24, 123, 129, 138, 149$
4	(4)	56	7	64	1,2,3,4,5,10	[1-6],8,9,10,**64**,149
5	(4)	90	100	-	3,6,8,11,14,17,18	[1-11],[14-21],26,39,44,60,**100** ,104,112,164,172,180,191
6	(5)	100	200	150	4,6,8,11,17,19	[1-11],[13-20],38,44,**100**, 112,120,123,144,172,180,185,191,197,**200**
7	(4)	90	100	-	6,8,17,19,100,112	[1-4],6,7,8,10,11,[14-20],34,38,39,51,60, **100**,112,113,120,123,164,168,172,180,189,191
8	(5)	100	5	200	3,6,8,11,14,17,18	[1-4],[6-11],[14-21],26,39,44,60,**100** ,104,112,164,172,180,191

Table 5. Results obtained ReliefF using different problems of interaction between features based on equations (4) and (5). F stands for the feature index and P stands for its position at the ranked list. Tests shown over the double line use 10 relevant features, while those below it use 20 relevant features.

Test	Eq.	f_1	f_2	f_3	Interaction						No Interaction					
					F	P	F	P	F	P	F	P	F	P	F	P
1	(5)	1	7	24	1	2	7	200	24	8	1	6	7	3	24	193
2	(5)	13	14	15	13	4	14	2	15	5	13	164	14	74	15	89
3	(5)	2	4	6	2	1	4	2	6	6	2	46	4	1	6	9
4	(4)	56	7	64	56	55	7	145	64	26	7	3	56	119	64	140
7	(4)	90	100	-	90	165	100	21	-	-	90	147	100	185	-	-

the interaction better than INTERACT (tests 5, 6 and 8), although it also returns a higher number of features. Test 7 suggests that the interaction term should have a high value in order to be considered for INTERACT.

As ReliefF returns a ranked list of features, the goal was to look for differences between both lists of ranked features, with and without interaction. This comparison is shown in Table 5. Similarly to the results of INTERACT and CFS, test 2 gets very good results, with the involved variables $\{x_{13}, x_{14}, x_{15}\}$ getting very high positions at the list, indicating their relevance. However, test 1 shows some poor results, because feature 7, included in three terms of the equation (4), gets the last position. Test 3 and 4 also denote a proper selection of the filter. Regarding the tests with R=20, some slight differences were observed in the lists, however they remarkable and only test 7 of table 4 is included in Table 5 to denote the increased value of feature 100, although feature 90 remains in a similar position.

Table 6. Results with different number of samples. %*success* indicates the number of relevant features placed at the first R features of the list. S is the subset of features that is returned by INTERACT and CFS, $|S|$ is its cardinality and %R stands for the percentage of relevant features included in S.

R	Number of samples	ReliefF %success	FCBF %success	INTERACT $\|S\|$	INTERACT R $\subset$ S (%)	CFS $\|S\|$	CFS R $\subset$ S (%)
10	100	20	30	5	20	14	50
	200	20	50	4	10	18	80
	400	40	100	8	80	9	90
	800	60	100	10	100	10	100
20	100	25	20	5	0	15	20
	200	20	50	6	25	22	50
	400	30	30	7	15	20	35
	800	60	85	14	65	20	90
30	100	13,3	26,7	4	3,3	9	6,7
	200	13,3	40	5	6,7	20	30
	400	3	70	6	20	35	73,3
	800	40	62,5	13	40	21	66,7
40	100	25	25	5	5	13	10
	20	15	32,5	5	5	33	32,5
	400	27,5	90	6	15	33	55
	800	32,5	90	7	17,5	34	30

Finally, table 6 includes the results obtained using different number of samples. ReliefF and FCBF returned a very different ranked list, and so a different threshold should be used for each. For example, fixing a threshold equal to 0.03 for FCBF will lead to 53 features in the first case ($R = 10$ and $m = 100$), while it will return an empty list with $R = 40$ and $m = 800$. Therefore, the %*success* was used instead of the threshold. The %*success* indicates the number of correct relevant features placed at the first R positions of the list. $R \subset S$ pinpoints the percentage of relevant features included in S and it is used for CFS and INTERACT. As it can be seen, CFS and FCBF present good performance results, and that FCBF gets poorer results than CFS when 40 relevant features are used. INTERACT exhibits a poor performance, but on the other hand it returns a very small set of features.

5 Conclusions

In this paper, four filter methods for feature selection are applied over a synthetic data set aimed at studying the performance of the methods regarding the number of features and samples, the level of noise and the interaction between features . All the methods work better with a reduced set of features, and the results get worse when the number of features increases. However, in the case of the CFS and FCBF, performance in the case of 40 relevant features is much better than for ReliefF. Regarding the level of noise, again all methods reduce their performance when the level of noise increases, but FCBF and CFS are the ones in which this worsening

is smoother, specially for CFS. Regarding the performance of the filter methods when considering interaction between features, ReliefF has not a solid behaviour, and CFS is again the one in which the results are more promising. Finally, CFS and FCBF exhibits an acceptable performance with a reduced set of samples. So, in summary, CFS is the filter method more adequate for our purposes, because it is the one that obtains better performance and maintains a smoother decrement of it when the number of features and the level of noise increment. Besides, it presents an adequate detection of interaction between features, a complicated aspect for filters, as can be seen in the results presented on the previous section. It can be argued that CFS is the method that has a higher cardinality, that is, detects more features than the real relevant ones. Although this is true, the number of features in excess is not high at all (around 7 more than needed, when noise is around 5-10%) and this is perfectly assumable when the filter is a previous step for eliminating irrelevant features, that will be followed by a wrapper method.

References

1. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.: Feature Extraction. In: Foundations and Applications, Springer, Heidelberg (2006)
2. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence Journal, Special issue on relevance 97(1-2), 273–324 (1997)
3. Liu, H., Dougherty, E., Gy, J.D., Torkkola, K., Tuv, E., Peng, H., Ding, C., Long, F., Berens, M., Yu, L., Forman, G.: Evolving feature selection. IEEE Intelligent systems 20, 64–76 (2005)
4. Kira, K., Rendell, L.: A practical approach to feature selection. In: Proceedings of the Ninth International Conference on Machine Learning, pp. 249–256 (1992)
5. Kononenko, I.: Estimating attributes: Analysis and extensions of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) Machine Learning: ECML-94. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
6. Robnik-Sikonja, M., Kononenko, I.: Theoretical and empirical analysis of ReliefF and RReliefF. Machine Learning 53, 23–69 (2003)
7. Hall, M.A.: Correlation-based Feature Selection for Machine Learning. PhD thesis, University of Waikato, Hamilton, New Zealand (1999)
8. Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: ICML. Proceedings of The Twentieth International Conference on Machine Learning, pp. 856–863 (2003)
9. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical recipes in C. Cambridge University Press, Cambridge (1988)
10. Zhao, Z., Liu, H.: Searching for interacting features. In: IJCAI. Proceedings of International Joint Conference on Artificial Intelligence, pp. 1156–1161 (2007)
11. Quevedo, J.R., Bahamonde, A., Luaces, O.: A simple and efficient method for variable ranking according to their usefulness for learning. Journal Computational Statistics and Data Analysis (in press, 2007)
12. Fayyad, U.M., Irani, K.B.: On the handling of continuous-valued attributes in decision tree generation. Machine Learning 8, 87–102 (1992)
13. WEKA Machine Learning Project. Last access (September 2007), http://www.cs.waikato.ac.nz/~ml/
14. Liu, H.: Searching for interacting features. Last access (September 2007), http://www.public.asu.edu/~huanliu/INTERACT/INTERACTsoftware.html

FPGA-Based Architecture for Computing Testors

Alejandro Rojas, René Cumplido, J. Ariel Carrasco-Ochoa,
Claudia Feregrino, and J. Francisco Martínez-Trinidad

Computer Science Department, INAOE, Apdo. Postal 51 &216
Tonantzintla, Puebla, México
{roherale, rcumplido, ariel, cferegrino, fmartine}@inaoep.mx

Abstract. Irreducible testors (also named typical testors) are a useful tool for feature selection in supervised classification problems with mixed incomplete data. However, the complexity of computing all irreducible testors of a training matrix has an exponential growth with respect to the number of columns in the matrix. For this reason different approaches like heuristic algorithms, parallel and distributed processing, have been developed. In this paper, we present the design and implementation of a custom architecture for BT algorithm, which allows computing testors from a given input matrix. The architectural design is based on a parallel approach that is suitable for high populated input matrixes. The architecture has been designed to deal with parallel processing of all matrix rows, automatic candidate generation, and can be configured for any size of matrix. The architecture is able to evaluate whether a feature subset is a testor of the matrix and to calculate the next candidate to be evaluated, in a single clock cycle. The architecture has been implemented on a Field Programmable Gate Array (FPGA) device. Results show that it provides significant performance improvements over a previously reported hardware implementation. Implementation results are presented and discussed.

Keywords: Feature Selection, Testor Theory, Hardware Architecture, FPGA.

1 Introduction

Although the theoretical aspect of computing irreducible testors is advanced, there are not practical hardware implementations reported previously, excepting a brute force approach [1]. The intensive computational requirements due to the exponential complexity of the algorithms can be met by a combination of technology improvements and efficient hardware architectures based on parallel computational models. Specific parallel architectures can be designed to exploit the parallelism found in the algorithms to speed up the processing. Further optimizations such as incremental processing and the use of multiple processing elements are also possible.

In Pattern Recognition, feature selection is a very important task for supervised classification. A useful way to do this selection is through Testor Theory. The concept of testor for Pattern Recognition was introduced by Zhuravlev [2] in 1966. He defined a testor as a subset of features that allows differentiating objects from different classes. Testors are quite useful, especially when an object description contains both

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 188–197, 2007.

qualitative and quantitative features, and maybe they are incomplete (mixed incomplete data)[3].

However, the algorithms used to compute all irreducible testors have exponential complexity which seriously limits their practical use. Since software implementations of these algorithms do not provide a reasonable performance for practical problems, an option is to migrate to hardware implementations based on programmable logic to take advantage of the benefits that they offer.

This work is a continuation of the work reported in [1] and reports the development of a configurable hardware architecture for computing testors using the BT algorithm. The architecture is based on a candidate generator that jumps over unnecessary candidates, thus reducing the number of comparisons needed.

The rest of the paper is organized as follows. Section 2 provides the theoretical foundation of testor identification and describes the BT algorithm. Section 3 presents the proposed hardware architecture. In section 4 the FPGA implementation and experimental results are presented. In section 5, the performance improvements are briefly discussed and the obtained results are compared against the brute force approach, and software implementation. Finally, section 6 presents the concluding remarks and directions for further research.

2 Algorithms for Computing Testors

Let *TM* be a training matrix with *K* objects described through *N* features of any type $(x_1,\ldots,x_N)$ and grouped in *r* classes. Let *DM* be a dissimilarity Boolean matrix (0=similar,1=dissimilar), obtained from feature by feature comparisons of every pair of objects from *T* belonging to different classes. *DM* has *N* columns and *M* rows, where $M>>K$.

Testors and Irreducible Testors are defined as follows:

Definition 1. A subset of features *T* is a testor if and only if when all features are eliminated, except those from *T*, there is not any row of *DM* with only 0´s.

Definition 2. A subset of features *T* is an irreducible testor if and only if *T* is a testor and there is not any other testor *T* ' such that $T' \subset T$.

In definition 1, if there is not any row of *DM* with only 0´s it means that there is not a pair of objects from different classes that are similar on all the features of *T*, that is, a testor *T* allows differentiating between objects from different classes.

The number of rows in *DM* could be too large, therefore a strategy to reduce this matrix without losing relevant information for computing irreducible testors was introduced [4].

Definition 3. If *t* and *p* are two rows of *DM*, then *p* is a *sub-row* of *t* if and only if:

a) *t* has 1 everywhere *p* has 1
b) there is at least one column such that *t* has 1 and *p* has 0

Definition 4. A row *t* of *DM* is a *basic row* of *DM* if and only if *DM* does not have any other row *t' such* that *t'* is a *sub-row* of *t*.

Definition 5. The matrix that contains only the *basic rows* of *DM* is called *basic matrix* and is denoted by *BM*.

Let *TT(M)* be the set of all irreducible testors of the Boolean matrix *M*, then [4]:

Proposition 1. *TT(DM)=TT(BM).*

This proposition indicates that the set of all irreducible testors calculated using *DM* or *BM* is the same. However, *BM* is smaller than *DM* and the construction of *BM* from *DM* is a very fast process, for example, the time for obtaining a *BM* matrix with 48 columns and 32 rows from a *DM* matrix with 48 columns and 193,753 rows, is about 0.21 seconds on a PC with an Intel Pentium 4 processor running at 3.0GHz with 2GB of RAM memory.

There are two kinds of algorithms for computing Irreducible Testors: the *internal scale algorithms* and the *external scale algorithms*. The former analyzes the matrix to find out some conditions that guarantee that a subset of features is an irreducible Testor. The latter looks for Irreducible Testors over the whole power set of features; algorithms that search from the empty set to the whole feature set are call *Bottom-Top* algorithms and algorithms that search from the whole feature set to the empty set are call *Top-Bottom* algorithms. The selected algorithm is a *Bottom-Top external scale algorithm*, called BT. In order to review all the search space, BT codifies the feature subsets as binary *N*-tuples where 0 indicates that the associated feature is not included and 1 indicates that the associated feature is included. For computing testors, BT follows the order induced by the binary natural numbers, this is, from the empty set to the whole feature set. The BT algorithm is as follows:

Step 1.- Generate first no null *N*-tuple $\alpha=(\alpha_1,\alpha_2,\ldots\alpha_N)=(0,\ldots,0,1)$.
Step 2.- Determine if the generated *N*-tuple α is a testor of *BM*.
Step 3.- If α is a testor of *BM*, store it and take $\alpha'=\alpha+2^{N-k}$ where *k* is the index of the last 1 in α.
Step 4.- If α is not a testor of *BM*, determine the first row ν of *BM* with only 0´s in the columns where α has 1´s and generate α' as:

$$\alpha'_j=\begin{cases}\alpha_j & if \quad j<k\\ 1 & if \quad j=k\\ 0 & if \quad j>k\end{cases},$$

where *k* is the index of the last 1 in ν.
Step 5.- Take $\alpha=\alpha'$
Step 6.- If α is not after (1,1,...,1,1) then, go to step 3
Step 7.- Eliminate from the stored testors those which are not irreducible testors.

Step 3 jumps over all the supersets that can be constructed from α by adding 1's (features) after the last 1 in α. For example if n=9 and α=(0,1,1,0,0,1,0,0,0) then k=6 and the following $2^{N-k}-1=2^{9-6}-1=7$ *N*-tuples represent supersets of the feature set represented by α, which is a testor, and therefore these supersets are testors but they are not irreducible testors, as it can be seen as follows:

α	feature set	
011001000	$\{x_2,x_3,x_6\}$	α
011001001	$\{x_2,x_3,x_6,x_9\}$	
011001010	$\{x_2,x_3,x_6,x_8\}$	
.....	.....	
011001111	$\{x_2,x_3,x_6,x_7,x_8,x_9\}$	
011010000	$\{x_2,x_3,x_5\}$	$\alpha'=\alpha+2^{N-k}$

} 7 non-irreducible testors

Step 4 jumps over all the sets that can not be a testor according to definition 1, because for any combination of 0's and 1's in those N-tuples, the row ν has 0's in those positions. For example if α=(011001001) and ν=(100100000) following step 4, the next N-tuple to be verified will be α'=(011100000) which has 1 in at least one position where ν has 1 (x_4). Note that all the N-tuples between α and the next N-tuple to be verified are not testors, because of ν, as it can be seen as follows:

ν	α	feature set	
100100000	011001001	$\{x_2,x_3,x_6,x_9\}$	α
	011001010	$\{x_2,x_3,x_6,x_8\}$	
	011001011	$\{x_2,x_3,x_6,x_8,x_9\}$	
	.....	.....	
	011011111	$\{x_2,x_3,x_5,x_6,x_7,x_8,x_9\}$	
	011100000	$\{x_2,x_3,x_4\}$	α'

} 31 non-testors

In step 4, the jump will be bigger if k is smaller. For this reason the columns and rows of *BM* are sorted is such way that the columns containing more 0´s are placed on the right and the rows with more 0´s are placed upper. It is important to remark that interchanging columns or rows will not affect the result of computing all irreducible testors, just the information about which column corresponds to which feature must be preserved [5], and that the process for reorganizing the rows and columns of *BM* is very fast.

3 Proposed Architecture

The process of deciding if an N-tuple is a testor of *BM* involves comparing the candidate against each one of the *BM*'s rows. For software-based implementations, this presents a big disadvantage, in particular for large matrices with many rows. The proposed hardware architecture exploits the parallelism inherent in the BT algorithm and evaluates whether a candidate is a testor or not in a single clock cycle. It is composed by two main modules as seen in Fig. 1. The BM module stores the input matrix and includes logic to decide if an input N-tuple is a testor. The candidate generator module produces the candidates to be evaluated by the BM module. To calculate the next candidate according to the BT algorithm, the architecture feedbacks the result of evaluating a candidate to the generator module that will generate the next candidate as specified by the BT algorithm. This process does not introduce latency, thus the architecture is capable of evaluating a candidate in a single clock cycle.

The BM module is composed of M sub-modules named V_x, as shown in Fig. 2. Each V_x module contains a row (N bits) of the *BM* matrix and logic to perform testor evaluation (Fig. 3). To decide if an N-tuple is a testor, a bitwise AND operation is performed between the constant stored in each V_x module and the current candidate.

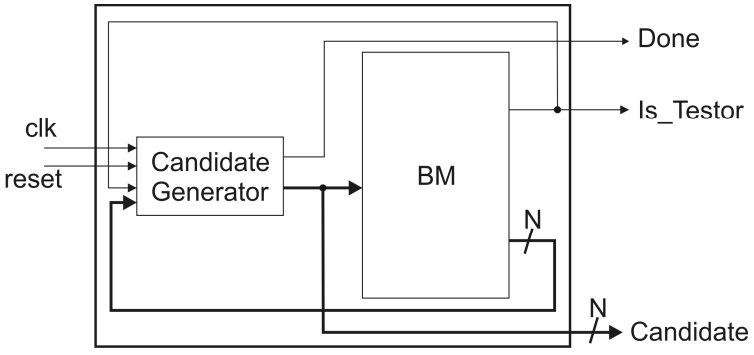


Fig. 1. Top-Level Architecture

If at least one bit of the AND operation result is TRUE, then the output *is_testor* of that particular V_x sub-module will be TRUE, and if the outputs of all V_x sub-modules are TRUE, then the output *is_testor* of the BM module will be TRUE, which means that the candidate is declared a testor of *BM*.

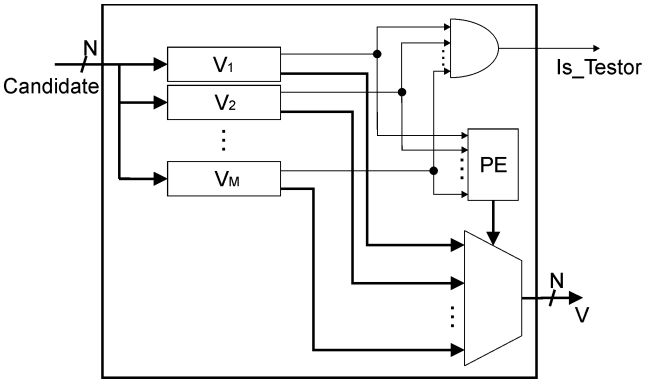


Fig. 2. BM module

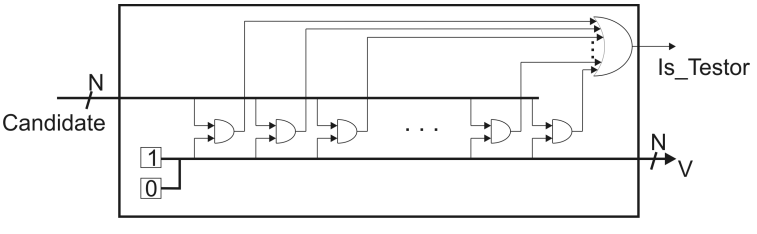


Fig. 3. Vx sub-module

When a candidate fails to be a testor of *BM*, the output *V* of the BM module contains the value of the row closest to the top that caused the test to fail. If the candidate is declared as testor, the output *V* is just ignored. The value of *V* is obtained

by using the output of a priority encoder as the select signal of a multiplexer that can select among all the rows of *BM*. This is similar to having a read address in a register file to access the value stored in a particular row.

The candidate generator module uses the feedback from the BM module to calculate the next candidate to be evaluated. As specified by the BT algorithm, there are two ways of generating the next candidate according to the evaluation result of the previous candidate. The candidate generator module consists of two sub-modules, the first sub-module (jump_1) generates the next candidate when the previous candidate is a testor and the second sub-module (jump_2) generates the next candidate when the previous candidate fails to be a testor. The next candidate is selected by a multiplexer according to the evaluation result of the previous candidate (Fig.4).

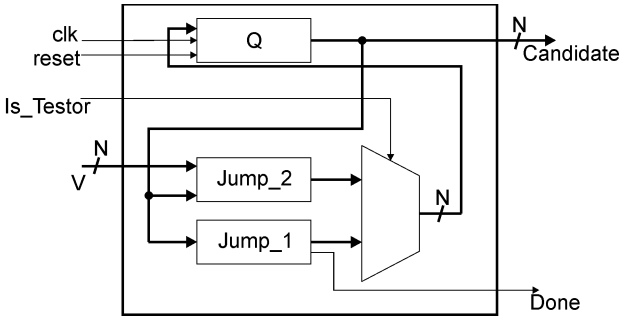


Fig. 4. Candidate generator module

Fig. 5 shows the jump_1 sub-module. It uses a priority encoder to obtain the index of the last '1' in the previous candidate value. The next candidate value is obtained by adding 2^{N-k} to the previous candidate as indicated by the step 3 of the BT algorithm.

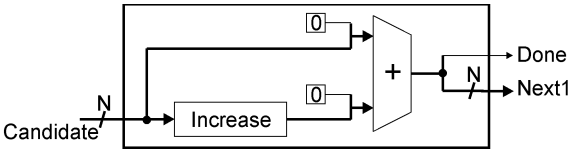


Fig. 5. Jump_1 sub-module

Fig. 6 shows the jump_2 sub-module. Besides the value of the previous candidate, it uses an input *V* that contains the value of the row of *BM* that caused the previous candidate not to be a testor. A priority decoder obtains the index *k* of the last '1' of *V*. By taking the value of the previous candidate, the next candidate is obtained by letting all bits to the left of the k^{th} position unchanged, the bits to the right are changed to '0', and the k^{th} bit is set to '1'. See step 4 of the algorithm.

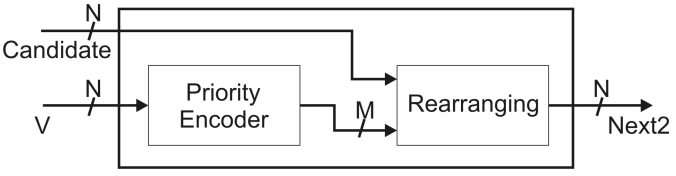


Fig. 6. Jump_2 sub-module

4 FPGA Implementation and Results

The proposed architecture was modeled in the VHDL Hardware Description Language under a structural approach. The VHDL model of the proposed architecture is fully parameterizable in terms of the matrix dimensions (*N*,*M*). The VHDL model was simulated and validated both functional and post-synthesis with ModelSim v6.0. The VHDL model was synthesized with Xilinx ISE v9.0 targeted for a medium size state-of-the-art Virtex-II Pro XC2VP30 FPGA device from Xilinx [6]. The use of the FPGA technology was chosen because it provides a rapid prototyping platform and is specially suited for implementing algorithms based on bit level operations.

The design was implemented on XtremeDSP Development Kit for Virtex-II Pro from Xilinx [7]. This board allows performing hardware-in-the-loop type of simulation using the PCI bus [8]. Although the synthesis results for all test cases show that the architecture can operate in excess of 50MHz, for the purpose of a fair comparison with the work reported in [1], the following results were calculated considering 50MHz as the operating frequency.

In order to show the performance of the proposed architecture, it was compared against the brute force implementation reported in [1], and software implementations of the BT algorithm and CT algorithms [9]. For experimentation purposes, basic matrices from 20 to 30 columns by 100 rows were randomly generated. Figure 7 shows the resulting processing times and Table 1 shows additionally the percentage of candidates tested by the BT algorithm.

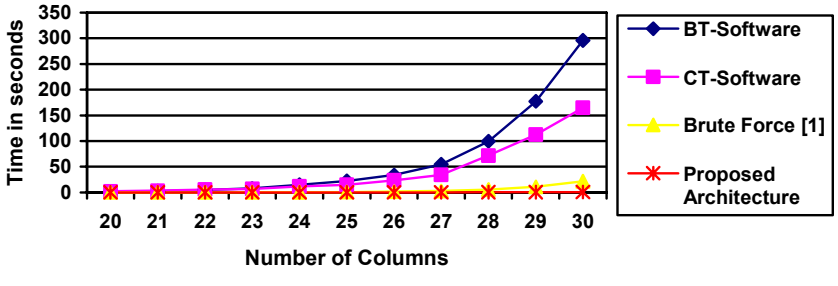


Fig. 7. Processing time in seconds for randomly generated data

Table 1. Processing times in seconds for different implementations of BT and CT algorithms

Matrix	Software implementations		Hardware implementations		Percentage of candidates tested - BT algorithm
	BT	CT	[1] (Brute Force)	Proposed Architecture	
20x100	1.55	2.06	0.021	0.004	17.79
21x100	2.23	3.16	0.042	0.006	14.59
22x100	4.73	5.11	0.084	0.009	11.38
23x100	7.61	6.72	0.168	0.016	9.54
24x100	14.50	11.55	0.335	0.027	7.94
25x100	22.56	14.78	0.671	0.047	7.08
26x100	33.80	23.43	1.342	0.079	5.86
27x100	54.68	34.14	2.684	0.140	5.20
28x100	99.34	71.75	5.369	0.209	3.88
29x100	177.25	112.33	10.737	0.316	2.94
30x100	295.92	164.84	21.475	0.517	2.41

The processing time t for a specific matrix is given by:

$$t = \left(\frac{2^N}{f} \right) \left(\frac{c}{100} \right)$$

where f is the clock frequency of the architecture and c is the percentage of candidates tested. Note that the value of c is data dependent, i.e. it varies according to each *BM* matrix.

These experiments show that the proposed architecture allows running BT 570 times faster that the software implementation and 318 times faster than software implementation of CT for N=30 and M=100. The improvement against the brute force for this matrix is 40X, this is because the percentage of candidates tested by BT is ~2.4%. The software implementations were executed on a PC with an Intel Pentium 4 processor running at 3.0GHz with 2GB of RAM memory.

The architecture has been designed to process variable sizes of the *BM* matrix. The maximum size of the matrix that can be implemented is only limited by the available resources on the target FPGA. The hardware resources utilization is proportional to the size of the matrix, i.e. the total number of elements NxM. Table 2 summarizes the FPGA resources utilization for large randomly generated matrices of 100 columns by 100 to 300 rows. Note that the number of columns was set to 100 as most of the practical problems that use testor theory will have at most this number of columns.

These results show that the hardware resources required to implement the architecture are proportional to the total number of elements in the *BM* matrix, e.g the 100x200 matrix requires twice as much slices than the 100x100 matrix. Note that even for the larger matrix (30,000 elements), the number of slices used is around 71% of the total available, which means that matrices up to around 40,000 elements can be processed with this modest size FPGA device. In this approach the matrix is declared as constant, which means that for any new matrix to be processed the architecture has

to be resynthesized and the FPGA configured. However, this process takes only a few extra seconds, but as no flip-flops are needed to store the matrix, the FPGA resources utilization is considerably reduced.

Table 2. Hardware resources utilization for large matrices

Matrix	Frec. (MHz)	Slices	Flip-Flops
100x100	87.73	3,391 (24%)	601 (2%)
100x150	82.37	5,064 (36%)	801 (2%)
100x200	80.49	6,567 (47%)	966 (3%)
100x250	75.99	8,060 (58%)	1,288 (4%)
100x300	77.00	9,778 (71%)	1,704 (6%)

5 Discussion

The proposed architecture provides higher processing performance than the previously reported hardware implementation as it now performs the complete BT algorithm. In spite of the added functionality, the architecture still is capable of performing the number of operations needed to test if an *N*-tuple is a testor of *BM* in a single clock cycle. Thus the performance improvement is directly related to the percentage of candidates tested (*c*), which in turn heavily depends on the values in the *BM* matrix. However, for real data this improvement could be significantly higher.

Experiments show that the proposed architecture allows computing testors faster than software implementations of the BT and CT algorithms, with improvements in the range of 2 orders of magnitude. However, for very large real data this improvement could be significantly higher. Additionally, an advantage of the proposed architecture is that it requires only one clock cycle to test each candidate independently of the number of rows, whereas software implementations processing time will significantly increase for matrices with a large number of rows.

It is important to highlight that the proposed architecture computes testors and the decision about which of them are irreductible has to be taken after each testor is found, this applies also to software-based implementations.

6 Conclusions

In this work, an efficient hardware implementation of the BT algorithm for computing testors is presented. The high performance of the proposed architecture is feasible due to the high level of parallelism implicit in the BT algorithm that can be efficiently implemented on a FPGA. The architecture is capable of evaluating a candidate in a single clock cycle for any *BM* matrix, regardless of the number of columns and rows, being the only limitation the size of the used FPGA device. The architecture provides a good trade-off between performance and hardware resource utilization and it is suitable to be used as a high performance processing module in a hardware-in-the-loop approach.

Even though the proposed architecture offers an improvement compared with a previously reported hardware implementation, further improvements, such as testing two or more candidates per iteration, are still possible. Also, because resource requirements are relatively small, a scheme where the processing core can be replicated will also be explored; this will effectively reduce the processing times proportionally to the number of processing cores that can be accommodated on the FPGA device. The final goal is to build a high performance flexible hardware/software platform for computing testors. On this direction, we are currently exploring the implementation of hardware architectures for more sophisticated algorithms like LEX [10].

References

1. Cumplido, R., Carrasco-Ochoa, A., Feregrino, C.: On the Design and Implementation of a High Performance Configurable Architecture for Testor Identification. In: Martínez-Trinidad, J.F., Carrasco Ochoa, J.A., Kittler, J. (eds.) CIARP 2006. LNCS, vol. 4225, pp. 665–673. Springer, Heidelberg (2006)
2. Dmitriev, A.N., Zhuravlev, Y.I., Krendeliev, F.P.: About Mathematical Principles of Objects and Phenomena Classification. Diskretni Analiz 7, 3–15 (1966) (in Russian)
3. Martínez-Trinidad, J.F., Guzmán-Arenas, A.: The Logical Combinatorial Approach to Pattern Recognition an Overview through Selected Works. Pattern Recognition 34(4), 741–751 (2001)
4. Lazo-Cortes, M., Ruiz-Shulcloper, J., Alba-Cabrera, E.: An Overview of the Evolution of the Concept of Testor. Pattern Recognition 34(4), 753–762 (2001)
5. Sánchez Díaz, G., Lazo Cortés, M.: Modifying BT Algorithm for Improving its Runtimes. Revista Ciencias Matemáticas 20(2), 129–136 (2002) (in Spanish)
6. Virtex-II Pro Data sheet, www.xilinx.com
7. XtremeDSP Development Kit for Virtex-II Pro, www.xilinx.com
8. Gomez, M.: Hardware-in-the-Loop Simulation. Embedded Systems Programming 14(13) (2001)
9. Bravo Martinez, A.: Algorithm CT for Calculating the Typical Testors of k-valued Matrix. Revista Ciencias Matematicas 4(2), 123–144 (1983) (in Spanish)
10. Santiesteban-Alganza, Y., Pons-Porrata, A.: LEX: A New Algorithm for Computing Typical Testors. Revista Ciencias Matemáticas 21(1), 85–95 (2003) (in Spanish)

Minimal BSDT Abstract Selectional Machines and Their Selectional and Computational Performance

Petro Gopych

Universal Power Systems USA-Ukraine LLC, 3 Kotsarskaya st., Kharkiv 61012, Ukraine
pmg@kharkov.com

Abstract. Turing machine (TM) theory constitutes the theoretical basis for contemporary digital (von Neumann) computers. But it is problematic whether it could be an adequate theory of brain functions (computations) because, as it is widely accepted, the brain is a selectional device with blurred bounds between the areas responsible for data processing, control, and behavior. In this paper, by analogy with TMs, the optimal decoding algorithm of recent binary signal detection theory (BSDT) is presented in the form of a minimal one-dimensional abstract selectional machine (ASM). The ASM's hypercomplexity is explicitly hypothesized, its optimal selectional and super-Turing computational performance is discussed. BSDT ASMs can contribute to a mathematically strict and biologically plausible theory of functional properties of the brain, mind/brain relations and super-Turing machines mimicking partially some cognitive abilities in animals and humans.

Keywords: neural networks, Turing machine, brain, memory, consciousness.

1 Introduction

It is widely accepted that no possible abstract [1] or physically realizing [2] computational device can be more powerful than Turing machine (TM) and this fact is actually the formal theoretical substantiation for the construction of contemporary digital (von Neumann) computers. But lately it has been speculated that in nature such physical processes may exist that TM computational abilities are insufficient for their simulations. The most famous hypothesis states that Turing computations are not strong enough to model human intelligence though there is no consensus of opinions on what human mind super-Turing abilities could actually mean [3,4,5].

On the other hand, it is widely accepted that the brain is a *selectional* device [6,7] which, in contrast to von Neumann computers, has blurred bounds between its hierarchically constructed and often overlapped functional areas and runs as a learnable system continuously adapting to its environment, not under the control of a given program. The brain demonstrates also its high tolerance to errors (noise) in data and damages to computational devices and can provide multiple correct solutions to a given problem. Additionally, the brain is profoundly influenced by the human genome that defines, to great extent, human dynamic behavior and cognitive abilities [7,8]. The traditional concept of information does not consider the meaning of information

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 198–208, 2007.

but in biology it may be clearly defined (that is what 'was selected') though, due to the multilevel brain hierarchy, this meaning is extremely difficult to specify [7].

In recent years, analog recurrent neural networks (ARNNs) were intensively studied as possible super-Turing abstract computational devices and it was demonstrated that with real-valued weights they can outperform standard TM in terms of its computational power [9]. ARNNs are in part motivated by the hypothesis that real neurons can hold continuous values with unbounded precision because of continuity of their underlying physical and chemical processes. Now the idea of analog brain computations continues to attract new adherents [10], though most experts doubt whether it is possible to implement analog devices with weights of unbounded precision (e.g., because of unavoidable noise, finite dynamic range, energy dissipation, or as 'the strength of the synapse is not very important, once it is large enough' [11]).

Taking into account that the brain is a selectional device, by analogy with TMs [1], we propose another approach in this paper: a brain-specific biologically relevant computational formalism of *abstract selectional machines* (ASMs), devices extracting from their input a given message (pattern or image). For this purpose, we transform optimal decoding algorithms of recent binary signal detection theory (BSDT) [12-17] into ASMs of different types and find their *optimal* selectional as well as *super-Turing* computational performance. The BSDT ASM's *hypercomplexity* is explicitly hypothesized; biological plausibility of ASMs, their parallels in TMs and some other abstract devices (e.g., [9, 18]) are discussed.

2 BSDT Coding/Decoding and Performance

The BSDT [12-17] operates with N-dimensional vectors x with their components $x^i = \pm 1$, a reference vector $x = x_0$ representing the information stored or that should be stored in a neural network (NN), binary noise $x = x_r$ (the signs of its components are randomly chosen with uniform probability, ½), and vectors $x(d)$ with components

$$x_i(d) = \begin{cases} x_0^i, & if \quad u_i = 0, \\ x_r^i, & if \quad u_i = 1 \end{cases} \quad d = \sum u_i / N, \quad i = 1,..., N \tag{1}$$

where u_i is 0 or 1. If m is the number of marks $u_i = 1$ then $d = m/N$, $0 \le d \le 1$; d is a fraction of noise components in $x(d)$, $q = 1 - d$ is a fraction of intact components of x_0 in $x(d)$ or an *intensity of the cue*, $0 \le q \le 1$. If $d = m/N$, the number of different $x(d)$ is $2^m C^N_m$, $C^N_m = N!/(N-m)!/m!$; if $0 \le d \le 1$, this number is $\sum 2^m C^N_m = 3^N$ ($m = 0, 1, \ldots, N$). As the set of $x(d)$ is complete, always $x = x(d)$.

The data coded as described are decoded by a two-layer NN with N model neurons in its entrance and exit layers which are linked by the rule 'all-entrance-layer-neurons-to-all-exit-layer-neurons.' Its synapse matrix elements are $w_{ij} = \xi x^i_0 x^j_0$ where $\xi > 0$ ($\xi = 1$ below), $w_{ij} = \pm 1$. That is a perfectly learned intact NN storing *one* reference pattern x_0 only. The NN's input $x = x_{in}$ is decoded (x_0 is identified in x_{in}) successfully if x_{in} is transformed into the NN's output $x_{out} = x_0$ (such an x_{in} is called a *successful input*, x_{succ}); an additional 'grandmother' neuron (an integrate-and-fire coincidence neuron responding to a precise combination of its inputs, x_0) checks this fact. The weighted sum of all inputs to the jth exit-layer neuron is $h_j = \sum w_{ij} x^i_{in}$ where x^i_{in} is an

input/output signal of the *i*th entrance-layer neuron, a fan-out that conveys its input to all exit-layer neurons. The output of the *j*th exit-layer neuron is

$$x^j_{out} = \begin{cases} +1, & if \quad h_j > \theta \\ -1, & if \quad h_j \le \theta \end{cases} \tag{2}$$

where $\theta \ge 0$ is the neuron's triggering threshold (for $\theta < 0$ see ref. 14), the value $x^j_{\text{out}} = -1$ at $h_j = \theta$ was arbitrary assigned. If $x^j_{\text{out}} = x^j_0$ ($j = 1, \ldots, N$) then x_{in} is x_0 damaged by noise; otherwise, it is a sample of noise, x_{r}. The above NN decoding algorithm can also be presented in functionally equivalent convolutional and Hamming distance forms each of which is the best in the sense of pattern recognition quality [12,14,17].

For intact perfectly learned NNs, decoding probability of vectors $x = x(d)$, $d = m/N$, can be calculated analytically [12,17]:

$$P(N,m,\Theta) = \sum_{k=0}^{K} C_k^m / 2^m, \; K_0 = \begin{cases} (N-\Theta-1)/2, & if \quad N \quad is \quad odd \\ (N-\Theta)/2-1, & if \quad N \quad is \quad even \end{cases} \tag{3}$$

where Θ is an even integer θ, $-N \le \Theta < N$; if $K < K_0$ then $K = m$ else $K = K_0$ (in this context k is the Hamming distance between x and x_0, K is its threshold value, and K_0 is the K for a given Θ). If $\Theta < -N$ then $P(N,m,\Theta) = 1$, if $\Theta \ge N$ then $P(N,m,\Theta) = 0$. For any $\theta \in \Delta\theta_j$, the NN decoding algorithm (see Eq. 2) gives $P(N,m,\theta) = P(N,m,\Theta_j)$ where $\Theta_j \in \Delta\theta_j$ (here, $j = 0, 1, 2, \ldots, N+1$, $\Theta_j = 2j - N - 1$). If $0 < j < N + 1$ then $\Delta\theta_j = [\Theta_j - 1, \Theta_j + 1)$ and $\Delta\theta_j = [\Theta_j, \Theta_j + 2)$ for odd and even N, respectively; if $j = 0$ and $j = N + 1$ then $\Delta\theta_0 = (-\infty, -N)$, $P(N,m,\Theta_0) = 1$ and $\Delta\theta_{N+1} = [N, +\infty)$, $P(N,m,\Theta_{N+1}) = 0$. As BSDT decoding algorithm exists in three equivalent forms, many of its parameters can be calculated one through the other for a given N, e.g.: m, d, and q or Q (a convolution of x and x_0, $Q = \sum x^i x^i_0$, $-N \le Q \le N$), ρ (correlation coefficient, $\rho = Q/N$), k (Hamming distance, $k = (N + Q)/2$), Θ, Θ_j, $\theta \in \Delta\theta_j$, F_j (false-alarm probability or the probability of identification of a noise vector $x = x_{\text{r}}$ as x_0), and j (confidence level of decisions) [13,16]. For this reason, decoding probability (Eq. 3) can be written in some equivalent forms (as functions of different sets of their parameters): $P(N,m,\Theta) = P(N,m,\theta) = P(N,m,\Theta_j) = P(N,d,j) = P(N,d,F) = P(N,q,j) = P(N,q,F)$ etc.

3 Minimal BSDT ASMs and Their Functions

The BSDT explicitly defines a finite set of optimally coded objects, $x = x(d)$, and a finite-sized tool for their optimal decoding, a learned NN. Consequently, it is suitable for solving two broad classes of practical problems: studying a set of x given the NN (data mining, e.g. [19]) and studying an NN with the set of x given (memory modeling, e.g. [15,17]). To match these problems better, BSDT optimal decoding algorithm can be presented in either a feedforward form (the case of data mining) or a cyclic form (the case of memory modeling). The former and the latter correspond to *passive* ASMs (Section 3.1) and *active* ASMs (Section 3.2). For simplicity, *one-dimensional* (dealing with one-dimensional inputs) ASMs will be only considered.

3.1 Minimal One-Dimensional BSDT Passive ASM

First, by analogy with TMs, we present BSDT decoding algorithm studying a set of vectors x given the NN, as a minimal one-dimensional BSDT *passive ASM* (PASM, Fig. 1A; cf. Fig. 2 of ref. 15). It consists of an N-channel scanner (box 1), the learned NN (box 2), and grandmother neuron (diamond 3). The PASM is also supplied by a *finite-length* one-dimensional *read-only* data tape divided into equal cells bearing binary signals only, +1 or −1. The tape is *movable* (with its drive outside the PASM) while the scanner stays still, that is '*passive*.'

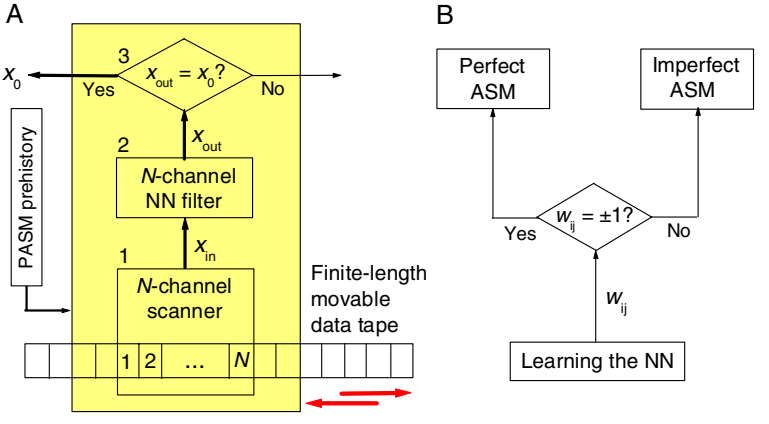


Fig. 1. Minimal one-dimensional BSDT PASM and conditions defining its selectional (classification) quality. **A**, The architecture of PASMs; arrows near the tape, possible directions of its movement; thick arrows, pathways for transferring vectors x (groups of N synchronous spikes); learned NN (box 2) plays the role of a filter tuned to select x_0; given its prehistory, each individual PASM can run in isolation. **B**, Influence of NN synaptic weights on ASM selectional performance ($w_{ij} = \pm 1$ and $w_{ij} = \pm\xi$, if $\xi > 0$, are functionally equivalent).

The scanner (box 1) reads out simultaneously N successive signs ±1 constituting together the PASM's input string x_{in}, transforms x_{in} into its active form (a set of N synchronous pulses or 'spikes' [17]), and conveys it to the learned NN (box 2). Here, x_{in} generates the NN's output x_{out} and, then, grandmother neuron (diamond 3) checks whether $x_{out} = x_0$. If that is the case, then $x_{in} = x_{succ}$, and x_0 becomes available for its further use in its active form. Afterwards, the tape shifts left or right[1], the scanner reads out next x_{in} and so forth, generally in a never-stop regime defined by the outside driver. *Feedforward* processing of each input x_{in} always gives the definite PASM selectional decision (that is the PASM's goal) during a finite time period, Δt_{PASM}.

[1] To write out 2^N different N-dimensional vectors x_{in} (strings of the size N of 1s and −1s), it is enough to have in general the tape of $2^N + N - 1$ cells. If so then these 2^N strings have to be read out in a definite manner only: for example, from left to right by shifting the tape in the scanner one cell left after each act of reading. If the tape of $2^N N$ cells is used then any x_{in} may be read out after the reading any other x_{in}.

All the PASM's internal connections as well as its environmental disposition were hardwired and all its parameters (N, x_0, w_{ij}, θ) were adjusted during its design process. This process (and its final product, a PASM) is defined by the PASM *prehistory* which, as we suppose, may be loosely divided into its evolutionary (genome-specific) and developmental (experience- or learning-specific) stages. If the environment changes then the PASM may be adapted (by outside factors) to new conditions but resulting PASM becomes already another one having already another prehistory.

A perfectly learned (with $w_{ij} = \pm 1$) BSDT NN produces selections the best in the sense of pattern recognition quality (Section 2). Hence, in contrast to ARNNs [9], for the construction of PASMs, NNs with rational or real weights are not required (in this case they lead to more complicate computations only). Of computational viewpoint a binary NN (as box 2 in Fig. 1A) is equivalent to a *finite automaton*, the simplest TM [9]. For verifying this NN's outputs the PASM uses the string x_0, a PASM specific *advice* common for all the inputs x_{in}. Consequently, the PASM may be considered as an advice TM [9] with advice sequence x_0 completely defined by PASM prehistory (the length of x_0, N, is the advice TM's noncomputability level [9]). If x_0 is known then any PASM, using it, may be simulated by a TM which, because any PASM is specified by its prehistory at a process level beforehand (as x_0 and a given pattern of NN connections), will spend for simulating the PASM selections the time $\Delta t_{TM} > \Delta t_{PASM}$. In that sense PASMs are *super-Turing* computational devices.

A contradiction arises: on the one hand, advice TMs define a nonuniform class of noncomputability [9] while, on the other hand, the same TMs substantiate the PASMs having super-Turing computational abilities. To resolve this contradiction, we recall that the tape does not contain any PASM instructions and the PASM's architecture, parameters, and advice were completely hardwired ('programmed') in the course of PASM prehistory, *before* the moment when the PASM was placed into operation (it is possible because all PASM inputs are finite-sized and their total amount is limited). Thus, the source of PASM hypercomputability is *decoupling* between programming and computations: all PASM computations are completely specified ('programmed') during the PASM prehistory which, in contrast to TM specifications, has uncertain length and may be in general of infinite length (not describable by finite means).

3.2 Minimal One-Dimensional BSDT Active ASM

Here, by analogy with TMs we present BSDT decoding algorithm studying an NN with the set of its inputs x given, as a minimal one-dimensional BSDT *active ASM* (AASM, Fig. 2; cf. Fig. 2 of ref. 15). It consists, in particular, of an N-channel movable (that is *'active'*) scanner (box 1), the learned NN (box 2), and grandmother neuron (diamond 3). The AASM is also supplied by an *infinite* one-dimensional still *read-only* data tape divided into equal cells bearing binary signals only, +1 or −1. This part of the AASM is actually a PASM (Fig. 1A) though AASMs additionally have the scanner's drive (box 6, see also footnote 1). Boxes 5 and 8 count NN inputs, diamonds 3 and 4 are the points of choice. It is supposed that internal loop (1-2-3-4-5-6-1) runs due to the scanner's *regular shifts* while external loop activation (1-2-3-4-7-8-6-1) leads to *a skip* of the scanner (in such a way any distant region on the tape becomes almost immediately available for the search; whether or not the skip is needed, its size and direction are defined by an *ASM society*, box 9).

To identify among AASM inputs x_{in} the one that is x_{succ}, the AASM successively examines each next x_{in} produced by one-step shift of the scanner. Once x_{succ} happens (i.e., $x_{out} = x_0$ in diamond 3), the search is finished, x_0 in its active form [17] is passed to ASM society (box 9), and the AASM is ready for searching for the next x_{succ}. If among i inputs produced by i successive regular scanner shifts there is no x_{succ} and i exceeds its upper bound[2], i_{max}, then diamond 4 interrupts internal loop and requests for *an advice* whether or not to continue the search (diamond 7). If the advice produced by ASM society (box 9, a counterpart to an 'oracle' [9] of TM theory) is 'to continue' then the scanner skips and the loop 1-2-3-4-5-6-1 is again activated.

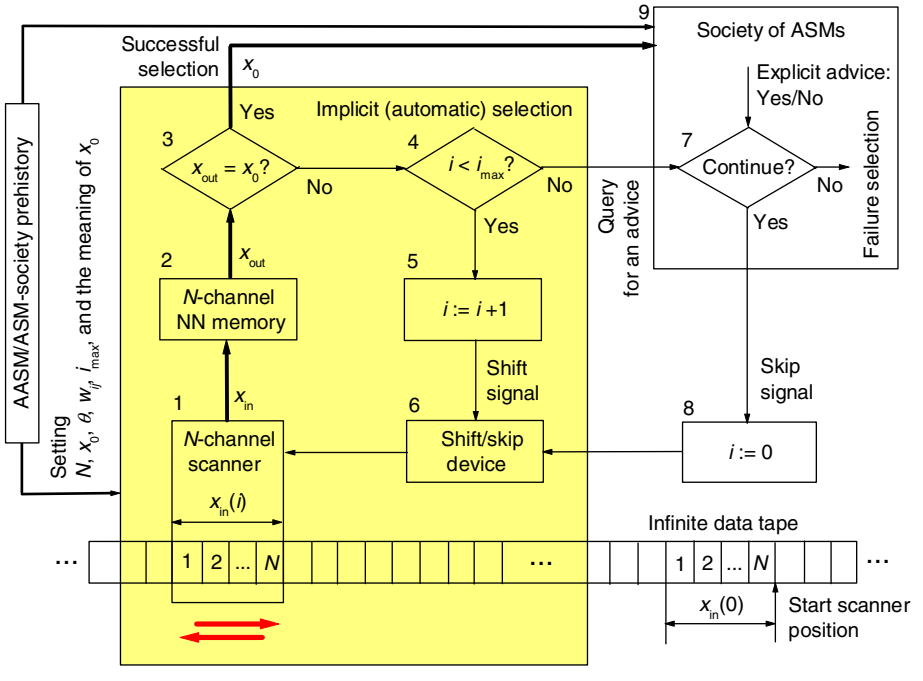


Fig. 2. The architecture of minimal one-dimensional BSDT AASMs. Arrows near the scanner, its possible movement directions; thick and thin arrows, pathways for vectors x and connections for transferring control asynchronous signals, respectively; learned NN (box 2) plays the role of memory unit storing the x_0; areas $x_{in}(0)$ and $x_{in}(i)$ on the tape are the initial and the ith input string, respectively; an individual AASM can run if it is only a member of the ASM society (a so far unspecified set of interactive ASMs equipped by sensory and executive devices, box 9).

Everything what concerns PASM selectional and computational performance (Section 3.1) is valid for the part of an AASM shaded in Fig. 2. But in spite of the AASM's optimality, no x_{succ} may be discovered, simply because the series of inputs consists of failure strings only (of x_{in} generating $x_{out} \neq x_0$). Hence, the selectional

[2] Parameter i_{max} is specified by AASM prehistory. For possible neurobiological reasons to fix i_{max}, see ref. 12, 15, 17; the BSDT natural value of i_{max} is the amount of different $x(d)$, 3^N.

process (by shifting the scanner) may never stop in general and the AASM's goal (discovering the x_0) may not be achieved in finite time — that is the *AASM halting problem*. To solve it, the search strategy should be sometimes changed according to an external advice. For AASMs, such an advice is produced by ASM society having a *hyperselectional* power, i.e., having the capability to generate the advice, in spite of its TM noncomputability, during a finite time period, Δt_{adv}. In reply to the query of diamond 4, in the milieu of ASM society, thanks to its collective properties and hypercomputational power of PASMs in AASM bodies, the advice is being selected (the halting problem is being solved) taking into account that it depends, we suppose, on the society's prehistory, its current assessment of rewards (successful selection) and punishments (failure selection), previous searching *history*[3], and time constraints.

The advice is the society history's 'event' produced by the society's current inputs and its current internal state given prehistory. On the other hand, this advice is also a part of particular AASM (pre)history and, consequently, the (pre)history of the whole society is simultaneously a part of (pre)history of its individual member, the AASM. Of this fact, a *hypercomplexity* of ASM/ASM-society (pre)histories follows: they are related, may span *infinitely* back in time and encompass events up to the origin of life (or even the Universe). Thus, the ASM society's hyperselectivity is ensured by its hypercomplexity: in the realm of ASMs their is no problem of hypercomputations but, instead, the problem of ASM/ASM-society hypercomplexity occurs.

4 Optimal Selectional Performance of Minimal BSDT ASMs

A PASM generates its definite selectional decision for each input, x_{in}. An AASM makes its selectional decisions when among its inputs a successful one, $x_{in} = x_{succ}$, is encountered. If PASMs and AASMs operate over the same set of their inputs then they have common selectional performance. For distinctness, below we define this performance using the set of $x_{in} = x(d)$ given a specific d or $0 \le d \le 1$ (Section2).

We introduce an ASM's *absolute* selectional power (SP) and *relative* SP: $\alpha(\theta,m/N)$ and $\gamma(\theta/N)$, respectively. $\alpha(\theta,m/N)$ is the amount of x_{succ} given θ and $d = m/N$: $\alpha(\theta,m/N) = P(N,m,\theta)2^m C^N_m$ where $P(N,m,\theta)$ is defined by Eq. 3 (see Section 2, Fig. 3A and its legend). $\gamma(\theta/N)$ is a fraction of x_{succ} given θ, N, and all d from the range $0 \le d \le 1$: $\gamma(\theta/N) = \beta(\theta/N)/\beta_{max}(N)$ where $\beta(\theta/N) = \sum\alpha(\theta,m/N)$, $\beta_{max}(N) = \sum 2^m C^N_m = 3^N$ and $m = 0, 1, 2, \ldots, N$ (Fig. 3B).

Fig. 3A demonstrates, in particular, that the largest number of x_{succ} is among inputs with an 'intermediate' m_0/N fraction of noise, though in this case the probability $P(N,m,\theta)$ of discovering the x_{succ} is not always maximal (e.g., in Fig. 3A, $m_0/N = 3/5 = 0.6$ while $P(5,3,2) = 1/2 < 1$). As Fig. 3B shows, larger $\gamma(\theta/N)$ and negative θ correspond to 'low-confidence' selections ($F > 1/2$) while smaller $\gamma(\theta/N)$ and positive

[3] We distinct prehistory and history. Prehistory is the process (and, eventually, the product) of designing the ASM and consists of its evolutionary (genome specific) and developmental (learning specific) stages (see also ref. 20). History is a series of ASM 'events' (outputs initiated by particular inputs) given its prehistory. Any PASM history is predictive because, given prehistory, it is completely defined by PASM inputs (Fig. 1A) while any AASM history is unpredictive and may infinitely rich become because, given prehistory, it depends on AASM inputs as well as the current state of ASM society (Fig. 2).

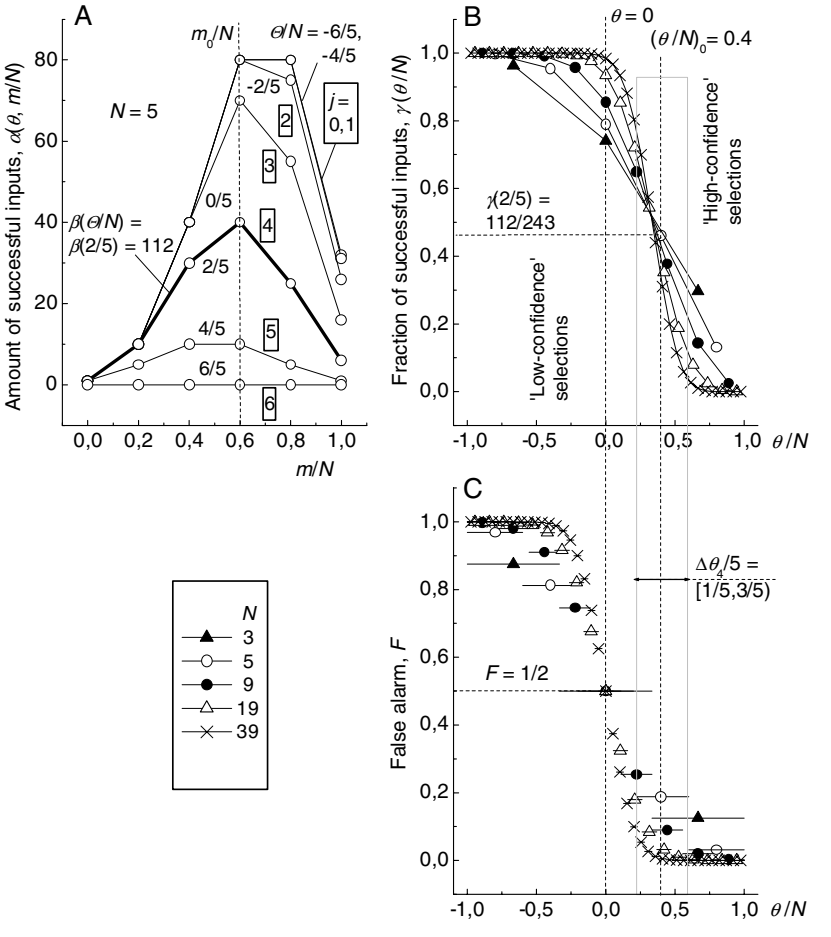


Fig. 3. Absolute (α) and relative (γ) selectional power for minimal one-dimensional BSDT ASMs. **A**, $\alpha(\theta,m/N)$, open circles connected by line segments (θ, neuron triggering threshold; m, the number of an input vector's noise components; N = 5); boxed numbers are decision confidence levels (the curves' serial numbers), j [16]; unboxed numbers designate middle points, Θ_j/N, of $\Delta\theta_j/N$ intervals in **C**; values of $\alpha(2,m/5)$ lie on the thick solid line (for them $\beta(\Theta_j/N) = \beta(2/5) = \sum\alpha(2,m/5) = 112$); dashed line depicts m_0/N, a fraction of input noise components at maximums of $\alpha(\theta,m/N)$. **B**, $\gamma(\theta/N) = \gamma(\Theta_j/N)$, different signs ($N$ = 3, 5, 9, 19, and 39; Θ_j/N, abscissa of each sign); horizontal dashed line points to that open circle whose $\gamma(\theta/N)$ corresponds to thick line in **A**, $\gamma(2/5) = \beta(2/5)/\beta_{max}(5)$, $\beta_{max}(5) = 3^5 = 243$; $\gamma(\theta/N < 1) = 1$ and $\gamma(\theta/N \geq 1) = 0$ are not shown. **C**, False alarms for relative neuron triggering threshold intervals, $\Delta\theta_j/N = [(\Theta_j - 1)/N, (\Theta_j + 1)/N)$; because of their infiniteness, $\Delta\theta_{N+1}/N$ and $\Delta\theta_0/N$ are not shown; vertical dashed lines indicate $\theta = \Theta_{(N+1)/2} = 0$ and the middle point, $(\theta/N)_0$, of the boxed interval, $\Delta\theta_4/5$, that can separate 'low-confidence' and 'high-confidence' selections. In all panels signs give the function's values for all $\theta/N \in \Delta\theta_j/N$, not for separate Θ_j/N only (cf. [13,14,16]).

θ correspond to 'high-confidence' selections ($F < 1/2$). This result is consistent with the fact that neuron thresholds $\theta \geq 0$ are preferable in practice. The region $-\infty < \theta/N < +\infty$ may also be divided into areas where the more the N the larger the $\gamma(\theta/N)$ is $[\theta/N < (\theta/N)_0]$ and the more the N the smaller the $\gamma(\theta/N)$ is $[\theta/N > (\theta/N)_0]$. Consequently, $(\theta/N)_0 \sim 0.4$ may be accepted as another definition of the border between 'low-confidence' and 'high-confidence' selections though this border has an N-dependent non-zero width, $\Delta\theta/N$ (see Fig. 3B and C).

5 Discussion and Conclusion

The BSDT ASM's input (a string of the size N, x_{in}) is an arbitrarily chosen finite fraction of in general infinite input data set, the sought-for output (x_0) is alsow fixed. Hence, any other NN with such properties may be considered as a kind of ASM, though no one (except BSDT ASMs) can be 'ideal' [14]. In particular, that concerns ARNNs [9] which are hypercomputational machines if real-valued weights are being used. In contrast, ASMs use integer weights (Fig. 1B) and, in spite of that, provide *both* optimal selectional performance (due to BSDT optimality) and super-Turing computational ability (due to decoupling between programming and computations).

ASM x_0-specificity and the notion of genome/learning-specific ASM prehistory allow to define *the meaning* of a message 'that was selected': the meaning of a successful input, x_{succ}, is the meaning of x_0 or the content of prehistory of the ASM selected x_0, $M(x_0)$. The total amount of x_{succ} may be large (Fig. 3A, B) but, for any x_{succ}, $M(x_{\text{succ}}) = M(x_0)$. This may explain why in biology *correct decisions are multiple and degenerate* (many different inputs, x_{succ}, produce the same output, x_0). $M(x_0)$ may exist in an implicit, $M_{\text{impl}}(x_0)$, or an explicit, $M_{\text{expl}}(x_0)$, form. $M_{\text{impl}}(x_0)$ is implemented as an ASM's hardwiring and parameter setting (see also [18, 20]), that is a '*subjective'* property not available out of the system. To be communicated to other *similar* (having partially the same prehistory) system, $M(x_0)$ *or its part* should be coded/described using a (natural) *language*—a code interpreted in computation theory, e.g. [9], as a discrete function defining word/meaning relations. The code/language description obtained is $M_{\text{expl}}(x_0)$. The more elaborate the language the more complete $M_{\text{expl}}(x_0)$ may be, $M_{\text{expl}}(x_0) \rightarrow M_{\text{impl}}(x_0)$ but never $M_{\text{expl}}(x_0) = M_{\text{impl}}(x_0)$; when a finite message is being coded/decoded, *the similarity* of communicating systems (of their prehistories) ensures the common *context* for them—an implicitly available and infinite in general additional information needed for unambiguous understanding the message.

For biological plausibility of AASMs considered as memory units, see ref. 15, 17. According to them, an AASM's internal loop is interpreted as a minimal structure representing *implicit (unconscious) memory* unit (it is shaded in Fig. 2) while together with the external loop it already represents an 'atom' of *explicit (conscious) memory* or an 'atom' of *consciousness* relying on an explicit ('conscious') advice. Biological plausibility of PASMs may be illustrated by BSDT theory for vision where 'a hierarchy of tuned local NN units' (i.e., PASMs) extracts 'step-by-step from an initial image its more and more general features/properties' [15, p. 149]. Also we draw attention to a clear analogy between shift and skip movements of an AASM's scanner and, respectively, slow drifts and saccades of the human eye [21].

In sum, an original 'selectional' approach to biologically plausible network computations has been introduced. BSDT ASMs were defined and it was demonstrated that they are hypercomplex hypercomputational learnable (not programmable) NN devices providing optimal selectional performance with preferably integer synaptic weights. This approach is a natural tool for the description of brain/mind functions (spike computations) and could contribute to the construction of mathematically strict and biologically relevant future theory of brain cognitive abilities and brain/mind relations (for the first explicit example, see ref. 16). Of the technical perspective, BSDT ASMs might substantiate a large and diverse family of original high-performance artifacts (complex super-Turing selectional machines constructed of minimal learned BSDT ASMs as their building blocks, sensory input devices and executive output devices) mimicking in part different perceptual and cognitive functions in animals and humans.

References

1. Turing, A.M.: On Computable Numbers with an Application to the Entscheidungs-problem. Proceed. London Math. Soc. 42, 230–265 (1936-1937)
2. Deutsch, D.: Quantum theory, the Church-Turing Principle and the Universal Quantum Computer. Proceed. Roy. Soc. London A 400, 96–117 (2005)
3. Searl, J.R.: Minds, Brains, and Programs. Behav. Brain Sci. 3, 417–457 (1980)
4. Penrose, R.: Shadows of Mind. Oxford University Press, Oxford, England (1994)
5. Copeland, B.B.J.: Beyond the Universal Turing Machine. Australasian J. Philosophy 77, 46–66 (1999)
6. Edelman, G.M.: Naturalizing Consciousness: A Theoretical Framework. Proceed. Natl. Acad. Sci. 100, 5520–5524 (2003)
7. Brent, R., Bruck, J.: Can Computers Help to Explain Biology? Nature 440, 416–417 (2006)
8. Butcher, L.M., Kennedy, J.K.J., Plomin, R.: Generalist Genes and Cognitive Neuroscience. Curr. Opin. Neurobiol. 16, 145–151 (2006)
9. Siegelmannn, H.T.: Neural Networks and Analog Computation: Beyond the Turing Limit. Birkhauser, Boston-Basel-Berlin (1999)
10. O'Reilly, R.C.: Biologically Based Computational Models of High-Level Cognition. Science 314, 91–94 (2006)
11. Mander, E., Goaillard, J.-M.: Variability, Compensation, and Homeostasis in Neuron and Network Function. Nature Rev. Neurosci. 7, 563–574 (2006)
12. Gopych, P.M.: ROC Curves within the Framework of Neural Network Assembly Memory Model: Some Analytic Results. Int. J. Inf. Theo. Appl. 10, 189–197 (2003)
13. Gopych, P.M.: Sensitivity and Bias within the Binary Signal Detection Theory, BSDT. Int. J. Inf. Theo. Appl. 11, 318–328 (2004)
14. Gopych, P.M.: Neural Network Computations with Negative Triggering Thresholds. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) ICANN 2005. LNCS, vol. 3696, pp. 223–228. Springer, Heidelberg (2005)
15. Gopych, P.M.: Generalization by Computation through Memory. Int. J. Inf. Theo. Appl. 13, 145–157 (2006)
16. Gopych, P.M.: Performance of BSDT Decoding Algorithms Based on Locally Damaged Neural Networks. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 199–206. Springer, Heidelberg (2006)

17. Gopych, P.M.: Foundations of the Neural Network Assembly Memory Model. In: Shannon, S. (ed.) Leading-Edge Computer Sciences, pp. 21–84. Nova Sci., New York (2006)
18. Dayan, P., Hinton, P.E., Neal, R.M., Zemel, R.: The Helmholtz Machine. Neural Computations 7, 889–904 (1995)
19. Gopych, P.M.: Identification of Peaks in Line Spectra Using the Algorithm Imitating the Neural Network Operation. Instr. Experim. Techn. 41, 341–346 (1998)
20. Mayfield, J.E.: Minimal History, a Theory of Plausible Explanation. Complexity 12, 48–53 (2007)
21. Martinez-Conde, S., Macknik, S.L., Hubel, D.H.: The Role of Fixational Eye Movements in Visual Perception. Nature Rev. Neurosci. 5, 229–240 (2004)

Active Learning for Regression Based on Query by Committee

Robert Burbidge*, Jem J. Rowland, and Ross D. King

Department of Computer Science, University of Wales, Aberystwyth, Penglais, Aberystwyth, SY23 3DB, UK

Abstract. We investigate a committee-based approach for active learning of real-valued functions. This is a variance-only strategy for selection of informative training data. As such it is shown to suffer when the model class is misspecified since the learner's bias is high. Conversely, the strategy outperforms passive selection when the model class is very expressive since active minimization of the variance avoids overfitting.

1 Introduction

In process control we might wish to identify the effect of factors such as temperature, pH, etc. on output but obtaining such information, for example by running the system at various temperatures, pHs, etc., may be costly. In query learning, our goal is to provide criteria that a learning algorithm can employ to improve its performance by actively selecting data that are most informative. Given a small initial sample such a criterion might indicate that the system be run at particular temperatures, pHs, etc. in order for the relationship between these controls and the output to be better characterized.

We focus on supervised learning. Many machine learning algorithms are passive in that they receive a set of labelled data and then estimate the relationship from these data. We investigate a committee-based approach for *actively* selecting instantiations of the input variables x that should be labelled and incorporated into the training set. We restrict ourselves to the case where the training set is augmented one data point at a time, and assume that an experiment to gain the label y for an instance x is costly but computation is cheap. We investigate under what circumstances committee-based active learning requires fewer queries than passive learning.

Query by committee (QBC) was proposed by Seung, Opper and Sompolinksy [1] for active learning of classification problems. A committee of learners is trained on the available labelled data by the Gibbs algorithm. This selects a hypothesis at random from those consistent with the currently labelled data. The next query is chosen as that on which the committee members have maximal disagreement. They considered two toy models with perfectly realizable

* Thanks to Hugh Mallinson for initial inspiration. This work is supported by EPSRC grant reference S47649.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 209–218, 2007.

targets. The algorithm was implemented in the *query filtering* paradigm; the learner is given access to a stream of inputs drawn at random from the input distribution. With a two-member committee, any input on which the committee members make opposite predictions causes maximal disagreement and its label is queried. It was shown under these conditions that generalization error decreases exponentially with the number of labelled examples, but for random queries (i.e. passive learning), generalization error only decreased with an inverse power law.

Freund et al. [2] showed that QBC is an efficient query algorithm for the perceptron concept class with distributions close to uniform. This provided a rigorous proof of the earlier results, along with some relaxations in the requirements. They suggested a reasonable heuristic for filtering of queries would be to select and label those inputs expected to reduce prediction error. They note that this could be applied when the labels were not binary or even discrete. This is related to the variance-based regression methods described below. For the QBC approach to work, there must be some disagreement over the committee. In the original work, this was achieved by means of a randomized algorithm. An alternative approach is to use different subsets of the data, as in query by bagging and query by boosting [3]. This is also the approach taken in the regression framework investigated below.

2 Active Learning of Real-Valued Functions

The aim in active learning of a real-valued function is to query the labels of inputs such that the generalization error is minimized. The expected error is:

$$\int_x E_T\left[(\hat{y}(x;\mathcal{D}) - y(x))^2|x\right] q(x)dx \ ,$$

where $E_T[\cdot]$ denotes expectation over $P(y|x)$ and over training sets $\mathcal{D}$, $q(x)$ is the input distribution and $\hat{y}(x;\mathcal{D})$ is the learner's output on input x, given the training set $\mathcal{D}$. The expectation in the integrand can be decomposed as [4]:

$$\begin{aligned} E_T\left[(\hat{y}(x;\mathcal{D}) - y(x))^2|x\right] = E\left[(y(x) - E[y|x])^2\right] + (E_{\mathcal{D}}[\hat{y}(x;\mathcal{D})] - E[y|x])^2 \\ + E_{\mathcal{D}}\left[(\hat{y}(x;\mathcal{D}) - E_{\mathcal{D}}[\hat{y}(x;\mathcal{D})])^2\right] \ , \end{aligned}$$

where $E_{\mathcal{D}}$ denotes expectation over training sets and the other expectations are with respect to the conditional density $P(y|x)$. The first term on the right hand side is the variance of y given x, i.e. the *noise*. The second term is the squared *bias*. The third term is the learner's *variance*. The bias and variance comprise the learner's mean squared error (MSE).

Assuming the learner is approximately unbiased, Cohn, Ghahramani and Jordan [5] propose to select inputs for labelling that minimize the variance of the learner. It is assumed that an estimate of $\sigma^2_{\hat{y}}$, the variance of the learner at x is available. For a new input $\tilde{x}$, an estimate of the learner's new variance at x given $\tilde{x}$ is computed. This requires an estimate of the distribution $P(\tilde{y}|\tilde{x})$, or, at least, estimates of its mean and variance. This gives an estimate of $\left\langle \tilde{\sigma}^2_{\hat{y}} \right\rangle$, the expected

variance of the learner at x after querying at $\tilde{x}$. This is integrated over the input space to give an estimate of the integrated average variance of the learner. In practice, a Monte Carlo approximation was used, $\left\langle \tilde{\sigma}^2_{\hat{y}} \right\rangle$ was evaluated at 64 *reference points* and $\tilde{x}$ chosen to minimize the average expected variance over the reference points. In [6], this approach was used to select data to train a neural network. In [5], it was extended to mixtures of Gaussians and to locally weighted regression. Active data selection based on minimizing variance was superior to random data selection for a toy 2-degree-of-freedom robot arm problem. This is a noisy problem where the target function is not perfectly realizable. Note, however, that this technique of selecting statistically 'optimal' training data cannot be applied to all machine learning algorithms.

Krogh and Vedelsby [7] considered committees of neural networks for learning real-valued functions. The committee consists of k networks and the output of network α on input x is $\hat{y}^\alpha(x)$. They defined the *ambiguity* at an input point $\tilde{x}$ as the variance in the predictions of the committee members:

$$\overline{a}(\tilde{x}) = \sum_{\alpha} \left(\hat{y}^\alpha(x) - \overline{y}(x)\right)^2 \quad .$$

This provided a reliable estimate of the generalization error of the committee, and to determine the optimal voting weights of the committee members in determining the committee's predictions. These two contributions of that work have subsequently been cited a number of times in the literature. However, a third aspect of the work that is rarely cited relates to active learning. They propose to query at each step the label of the input for which the ambiguity is maximal, i.e. where the committee's variance is highest. This can be seen to be a minimax approach to the problem of minimizing the learner's variance over the input distribution. The networks were trained on the same set of labelled examples, starting from one labelled example and adding one labelled example at a time. The disagreement in the predictions of the individual committee members arises from the differing random initializations of the network weights. A committee of five neural networks, each with 20 hidden nodes, was trained to approximate the univariate square wave function. This is a noise-free problem where the target function is not realizable. Active selection of training data led to improved performance compared to random selection.

RayChaudhuri and Hamey [8] used a similar approach to [7]. However, the disagreement among committee members arose from their being trained on different sub-samples of the available data. An initial random sample of ten points was queried. Ten neural networks were each trained on random sub-samples of half the data. An unlabelled input with maximal variance over the committee is then selected. This approach is similar to query by bagging used in classification [3]. It was argued that this approach should lead to better performance than [7]. The target function was generated by a univariate neural network with three hidden nodes. The committee members all had the same architecture. The emphasis was not to minimize generalization error, but to minimize data collection. Active learning was shown to require fewer queries than passive learning to reach

agreement among the committee members, but its advantage was reduced when a small amount of noise was added (signal-to-noise ratio (SNR) ca. 40). Passive learning was superior when more noise was added (SNR ca. 2.5). In [9], a committee of five neural networks with one hidden node was used to approximate the step function using the criterion of [8]. Limited evidence was provided that their approach outperformed that of [7]. In [10], it was demonstrated that this active learning approach could be used for system identification by training a neural network on all of the data actively selected and using it as a feed-forward controller. In [11], RayChaudhuri and Hamey propose a similar criterion to estimate the variance of a learner at x. Instead of ambiguity they use a jackknifed [12] estimate of the variance combined with a noise estimate. Generalization error when using this selection criterion is not significantly different from that obtained using ambiguity for the problems considered here.

All of the criteria defined above were proposed within the framework of *selective sampling*. It is assumed that a set of unlabelled inputs is provided and we wish to query the labels of as few inputs as possible whilst minimizing the generalization error. In the toy problems considered in [5,6,7,8,9,10,11], it was assumed that the input distribution, $q(x)$, was uniform, and that the label of any input point, $\tilde{x}$, drawn from $q(x)$ could be queried. There are thus two basic approaches to optimizing the query criterion. As suggested in [5], the criterion could be optimized by hillclimbing on $\partial \left\langle \tilde{\sigma}^2_{\hat{y}} \right\rangle / \partial \tilde{x}$. The same idea could be applied to the other two criteria, by using a gradient free search to find a local maximum. In practice, in low dimensions, it is computationally more efficient to draw m candidate points from $q(x)$ at each iteration and choose the best $\tilde{x}$ from these. This is known as *pool-based* selective sampling. For example, [5] choose $m = 64$, [7] choose $m = 800$ and [8] choose $m = 100$. In situations where active learning outperforms passive learning, we would expect a large m to be beneficial. Conversely, in situations where active learning performs badly, a large m could lead to a substantial deterioration in performance[1].

Use of a committee to estimate the variance of a learner as a query criterion for active learning does not appear to have been pursued further than the foregoing references[2]. The aim of this paper is to investigate the performance of these criteria under various conditions.

3 Numerical Examples

The passive (P) selection strategy selects the next input point, $\tilde{x}$ at random from the unlabelled examples. For the active strategy, a committee of $k = 5$ learners is maintained. Each is trained on a subset of the labelled data by leaving out disjoint subsets of size $\lfloor n/k \rfloor$, where n is the number of labelled data. The

[1] Expressed in the saying 'give 'em enough rope and they'll hang themselves.'

[2] According to CiteSeer (`http://citeseer.ist.psu.edu/`), apart from the work by RayChaudhuri and Hamey, [7] has only been cited once with reference to active learning, viz. [13]. The work by RayChaudhuri and Hamey has also only been cited once, viz. [14]. Neither of these later works took a committee-based approach.

ambiguity (A) selection strategy selects $\tilde{x}$ to maximize $\overline{a}(x)$, $\tilde{x}$ is chosen from a pool of $m = 1000$ unlabelled examples. Results are averaged over 1000 runs.

We consider the toy 1-d problem described in [15]. The input dimension is $d = 1$ and the target function is:

$$f(x) = 1 - x + x^2 + \delta r(x) \ .$$

where

$$r(x) = \frac{z^3 - 3z}{\sqrt{6}} \text{ with } z = \frac{x - 0.2}{0.4} \ .$$

The number of queries is $n = 100$ and the labels are corrupted by i.i.d. noise $\sim N(0, 0.3^2)$. The test input density is $q(x) = N(0.2, 0.4^2)$ and is assumed to be known. The model class is linear regression with a polynomial kernel of order two. Three cases are considered, $\delta = 0, 0.005, 0.05$, termed *correctly specified*, *approximately correct*, and *misspecified*, respectively.

The signal-to-noise (SNR) ratio for this problem is $0.4^2/0.3^2 = 1.8$. This is roughly the same as the 'highly noise' problem in [8]. It was observed therein that ambiguity criterion did not outperform a passive selection strategy in the 'highly noisy' case. Following [8], we also consider low-noise, $\sigma_\epsilon = 0.1$ and zero-noise, $\sigma_\epsilon = 0.0$, versions of the same problem.

High Noise. Box-plots of the generalization error for the high noise case are in figure 1 (*top*). The mean and standard deviation of the generalization error are given in table 1. The passive committee learner does not have significantly different generalization error than reported in [15]. For the correctly specified and approximately correct cases, active selection does not have significantly different performance to passive selection. For the misspecified case, active selection performs significantly worse than passive learning. We do not necessarily expect the active strategy to work well in this case since the assumption that the bias is approximately zero has been violated.

Low Noise. Box-plots of the generalization error for the low noise case are in figure 1 (*middle*). The mean and standard deviation of the generalization error are given in table 2. There are no significant differences in performance between the passive and active strategies for the correctly specified case. In the misspecified and approximately correct cases, the performance of the active strategy is significantly worse than that of the passive strategy.

Table 1. The mean and standard deviation of the generalization error in the high-noise case. The best method and comparable ones by the t-test at 95% are emphasized in bold. All values in the table are multiplied by 10^3.

	$\delta = 0$	$\delta = 0.005$	$\delta = 0.05$
P	**2.86±2.35**	**2.90±2.62**	**6.11±3.44**
A	**2.74±4.24**	**3.09±4.45**	46.5±14.6

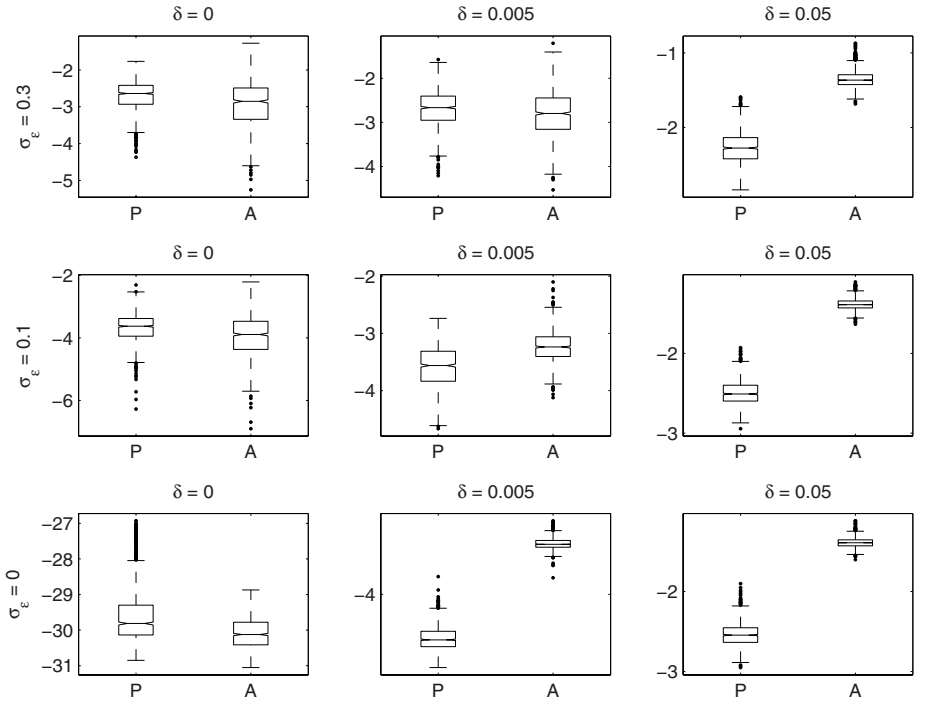


Fig. 1. Box-plots of the generalization error for the passive (P) and ambiguity (A) selection strategies. The box-plots show the distribution of $\log_{10}$ of the mean squared error on 1000 test points. The notches indicate robust estimates of the median generalization error.

Zero Noise. Box-plots of the generalization error for the zero noise case are given in figure 1 (*bottom*). The mean and standard deviation of the generalization error are given in table 3.

Active learning outperforms the passive strategy in the correctly specified case. This confirms the result of [8]. In the approximately correct and misspecified cases the active learning strategy is significantly worse than the passive strategy. This result is at odds with that reported in [7], although they used

Table 2. The mean and standard deviation of the generalization error in the low-noise case. The best method and comparable ones by the t-test at 95% are emphasized in bold. All values in the table are multiplied by 10^3.

	$\delta = 0$	$\delta = 0.005$	$\delta = 0.05$
P	**0.31±0.33**	**0.36±0.28**	**3.44±1.35**
A	**0.30±0.48**	0.76±0.64	41.8±7.34

Table 3. The mean and standard deviation of the generalization error in the zero-noise case. The best method and comparable ones by the t-test at 95% are emphasized in bold. All values in the table are multiplied by 10^3, except the values for $\delta = 0$, which are multiplied by 10^{30}.

	$\delta = 0$	$\delta = 0.005$	$\delta = 0.05$
P	42.5±139	**0.03±0.01**	**3.11±1.24**
A	**1.35±1.65**	0.42±0.07	41.3±6.18

a different method to create diversity across the ensemble. The generalization performance actually *worsened* as more points were queried.

In summary, active learning outperforms the passive strategy when the model class is correctly specified and there is no output noise. This performance gain is lost when the outputs are noisy or the model class is misspecified. Further consideration of this issue is given below.

3.1 Discussion

The problems of stagnation and deterioration in performance for the active learning strategy could be avoided by tracking the variance. This could be estimated from the working set, as in [8], or from a separate reference set, as in [5]. If the variance doesn't decrease then training can be halted, or the learner can switch to a passive strategy. A stochastic approach could also be taken, whereby the learner chooses an active or passive strategy probabilistically based on the change in average variance induced by the previous query. We do not investigate these ideas further but consider why the active strategy does not perform well.

In active learning the input density, $p(x)$, of training points differs from the input density, $q(x)$, of unlabelled points. This is known as the *covariate shift*. When $\delta = 0, \sigma_\epsilon = 0$, the active strategy outperform the passive strategy and empirically $p(x)$ is as shown in the left of figure 2. For the other cases, the active strategy does not outperform the passive strategy and $p(x)$ is as shown in the right of figure 2. In the previous work [7,8,9,10] $q(x)$ was taken to be uniform. This is likely to be the case in system identification and control [10]. We repeated the above experiments with $q(x)$ uniform having the same mean and standard deviation. Passive learning outperformed the active strategy in all cases.

When the model used for learning is correctly specified, the covariate shift does not matter since ordinary least squares (OLS) regression is unbiased under a mild condition [15]. In this case the median generalization error[3] of (A) is significantly lower than that of (P). When the model is misspecified, OLS is no longer unbiased. The active strategy investigated here is a variance-only method so we do not necessarily expect it to perform well when $\delta > 0$ and it does not. It is known that a form of weighted least squares is asymptotically unbiased for

[3] The expected generalization error of (A) is not significantly different from that of (P) since the active strategy is more variable in performance (see tables 1–3).

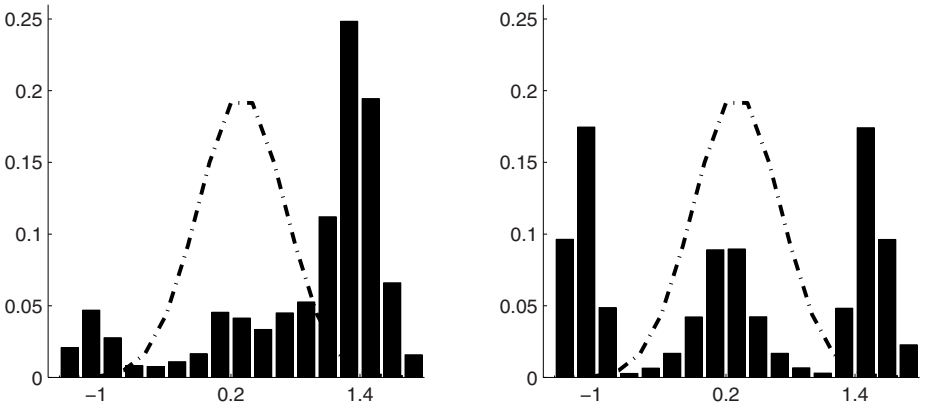


Fig. 2. Empirical input density, $p(x)$, for the active strategy. For the correctly specified, zero-noise case (*left*), the active strategy outperform the passive strategy. For the other cases (*right*), the active strategy perform the same or worse as the passive strategy. The test input distribution, $q(x)$, is superimposed.

misspecifed models [16]. Each training input point, x_i, is weighted by $q(x_i)/p(x_i)$. This requires an estimate of $p(x)$. We repeated the above experiments, at each iteration estimating $p(x)$ and using weighted least squares. The density $p(x)$ was estimated as $N(\mu, \sigma)$ where μ and σ are the maximum likelihood estimates. This did not improve the performance of the active strategy. It would be preferable to estimate $p(x)$ as a mixture of Gaussians, but this is not reliable with so few data, especially in higher dimensions.

For most real-life problems in function estimation and systems control, the target function or system is noisy and not perfectly realizable. When the model class is not correctly specified, this variance-only active learning strategy seems at best useless and at worst counterproductive. We now consider a different scenario and outline how to avoid the above problems.

When the model class is correctly specified, the active strategy is not significantly better than passive learning. When the model class is not correctly specified, the active strategy may be worse than passive learning. However, we have only considered cases where the model class is underspecified, i.e. learning tasks for which the learner is underfitting the available data. In most real-world situations, the functional form of the system is unknown. Ideally, we would use a very expressive model class in order to approximate the system as closely as possible. For example, it is known that a neural network with enough hidden nodes is able to approximate any function to arbitrary accuracy. This is not possible with noisy data since it will lead to overfitting. Overfitting occurs when the model variance is high [17]. Therefore, one way to avoid overfitting is to actively minimize the variance of the learner. This is exactly the approach taken by the active learning strategy considered here. Moreover, when the model class is very expressive the learner's bias is small.

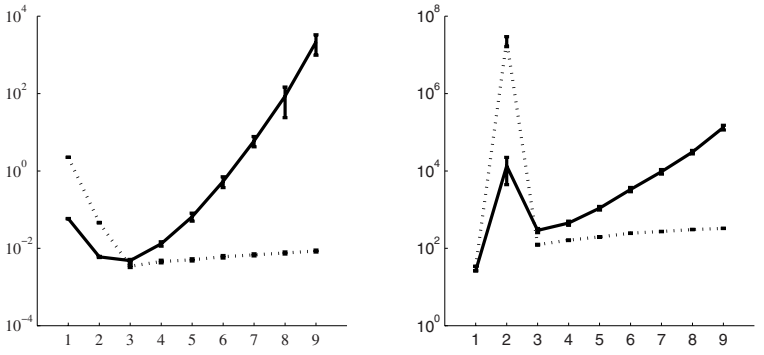


Fig. 3. Generalization error after 100 queries for the passive (*solid line*) and ambiguity (*dotted line*) selection strategies with a polynomial kernel as a function of the order of the polynomial for the toy data set (*left*) and the Boston housing data set (*right*). The error bars indicate 95% confidence limits for the mean.

In the following we consider only the misspecified, highly noisy case as this is the most realistic. The generalization error of (P) and (A) with polynomial kernels of order $\theta = 1, \ldots, 9$ is illustrated in figure 3 (*left*). When $\theta = 1, 2$, the learner underfits and (P) has lower generalization error than (A). As θ increases the model class becomes more expressive and performance of (P) deteriorates. However, (A) is robust to overspecification and has lower generalization error than (P) for $\theta \geq 4$. The lowest generalization error after 100 queries is for $\theta = 3$.

The Boston housing data set [18] has 506 examples and 13 attributes. The passive and active strategies were used to query the label of 100 points with polynomial kernels of orders $\theta = 1, \ldots, 9$. The generalization error averaged over 1000 runs is shown in figure 3 (*right*). The lowest error is at $\theta = 1$, when the passive strategy slightly outperforms the active. However, again the active strategy is much more robust to overspecification of the model class.

4 Conclusion

We have investigated a seemingly forgotten strategy for active selection of training data in real-valued function estimation. The main idea is to train a committee of learners and query the labels of input points where the committee's predictions differ, thus minimizing the variance of the learner by training on input points where variance is largest. This approach only works when the learner's bias is small. Its main advantage is that it is more robust to overspecification of the model order and thus less prone to overfitting than the passive strategy. The attractiveness of a committee-based approach is that there are no restrictions on the form of the committee members provided they are not identical.

References

1. Seung, H., Opper, M., Sompolinsky, H.: Query by committee. In: Proceedings of the Fifth Workshop on Computational Learning Theory, pp. 287–294. Morgan Kaufmann, San Mateo, CA (1992)
2. Freund, Y., Seung, H., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. Machine Learning 28(2–3), 133–168 (1997)
3. Abe, N., Mamitsuka, H.: Query learning strategies using boosting and bagging, pp. 1–9. Morgan Kaufmann, San Francisco, CA (1998)
4. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Computation 4, 1–58 (1992)
5. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. Journal of Artificial Intelligence Research 4, 129–145 (1996)
6. Cohn, D.: Neural network exploration using optimal experiment design. Morgan Kaufmann, Denver, CO (1996)
7. Krogh, A., Vedelsby, J.: Neural network ensembles, cross validation, and active learning. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) Advances in Neural Information Processing Systems 7, pp. 231–238. MIT Press, Denver, CO (1995)
8. RayChaudhuri, T., Hamey, L.G.: Minimisation of data collection by active learning. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, Australia, vol. 3, pp. 1338–1341 (1995)
9. RayChaudhuri, T., Hamey, L.G.: Accurate modelling with minimised data collection — an active learning algorithm. In: Proceedings of the Seventh Australian Conference on Neural Networks, pp. 11–15. Australian National University, Canberra, Australia (1996)
10. RayChaudhuri, T., Hamey, L.G.: Active learning for nonlinear system identification and control. In: Gertler, J., Cruz, J.J.B., Peshkin, M. (eds.) Proceedings of 13th International Federation of Automatic Control World Congress, vol. F, pp. 193–197. Pergamon, San Francisco, CA (1996)
11. RayChaudhuri, T., Hamey, L.G.: Cost-effective querying leading to dual control. Technical Report 96-07, School of MPCE, Macquarie University, New South Wales 2109, Australia (June 1996)
12. Quenouille, M.: Approximate tests of correlation in time series. Journal of the Royal Statistical Society B 11, 68–84 (1949)
13. Zhang, C., Chen, T.: An active learning framework for content-based information retrieval. IEEE Transactions on multimedia 4(2), 260–268 (2002)
14. Lindenbaum, M., Markovitch, S., Rusakov, D.: Selective sampling for nearest neighbor classifiers. Machine Learning 54(2), 125–152 (2004)
15. Sugiyama, M.: Active learning in approximately linear regression based on conditional expectation of generalization error. Journal of Machine Learning Research 7, 141–166 (2006)
16. Weins, D.: Robust weights and designs for biased regression models: least squares and generalized M-estimation. Journal of Statistical Planning and Inference 83(2), 395–412 (2000)
17. Bishop, C.: Neural Networks for Pattern Recognition. Clarendon Press, Oxford (1995)
18. Harrison, D., Rubinfeld, D.: Hedonic prices and the demand for clean air. J. Environ. Economics and Management 5, 81–102 (1978)

Influence of Wavelet Frequency and Orientation in an SVM-Based Parallel Gabor PCA Face Verification System

Ángel Serrano[1], Isaac Martín de Diego[1], Cristina Conde[1], Enrique Cabello[1], Linlin Shen[2], and Li Bai[3]

[1] Face Recognition & Artificial Vision Group, Universidad Rey Juan Carlos, Camino del Molino s/n, Fuenlabrada E-28943 (Madrid), Spain
{angel.serrano, isaac.martin, cristina.conde, enrique.cabello}@urjc.es
http://www.frav.es/

[2] Faculty of Information and Engineering, Shenzhen University, Shenzhen, 518060, China
llshen@szu.edu.cn

[3] School of Computer Science and IT, University of Nottingham, Nottingham, NG8 1BB, United Kingdom
bai@cs.nott.ac.uk

Abstract. We present a face verification system using Parallel Gabor Principal Component Analysis (PGPCA) and fusion of Support Vector Machines (SVM) scores. The algorithm has been tested on two databases: XM2VTS (frontal images with frontal or lateral illumination) and FRAV2D (frontal images with diffuse or zenithal illumination, varying poses and occlusions). Our method outperforms others when fewer PCA coefficients are kept. It also has the lowest equal error rate (EER) in experiments using frontal images with occlusions. We have also studied the influence of wavelet frequency and orientation on the EER in a one-Gabor PCA. The high frequency wavelets are able to extract more discriminant information compared to the low frequency wavelets. Moreover, as a general rule, oblique wavelets produce a lower EER compared to horizontal or vertical wavelets. Results also suggest that the optimal wavelet orientation coincides with the illumination gradient.

Keywords: Face Verification, Gabor Wavelet, Parallel Gabor Principal Component Analysis, Support Vector Machine, Data Fusion.

1 Introduction

Automated face recognition systems are developing rapidly, due to increasing computational capabilities, both in speed and storage, and its ease for use compared to other biometrics where the user collaboration is mandatory [1, 2]. There are a large variety of methods available in the literature for face recognition, such as Principal Component Analysis (PCA) [3] or Linear Discriminant Analysis (LDA) [4]. Some methods make use of Gabor wavelets [5] due to their similarities in behaviour to the human cells in the visual cortex. Following the standard definition given by [6] and [7], a Gabor wavelet is a 2D filter defined as a complex wave with a Gaussian

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 219–228, 2007.

envelope (Figure 1). It can be parameterized by a frequency ν (0≤ν≤4) and an orientation μ (0≤μ≤7):

$$\psi(\vec{r}) = \frac{k_\nu^2}{\sigma^2} \exp\left(-\frac{k_\nu^2 \|\vec{r}\|^2}{2\sigma^2} \right) \left[\exp\left(i\vec{k}_{\mu\nu} \cdot \vec{r} \right) - \exp\left(-\frac{\sigma^2}{2} \right) \right], \quad (1)$$

where $\vec{r} = (x, y)$ and $\sigma = 2\pi$. The wave vector, which determines the direction of the propagation of the wave, is defined as $\vec{k}_{\mu\nu} = k_\nu \left(\cos\varphi_\mu, \sin\varphi_\mu \right)$ with $k_\nu = 2^{(-(\nu+2)/2)}\pi$ and $\varphi_\mu = \mu\pi/8$ radians (with respect to the horizontal axis). $\vec{k}_{\mu\nu}$ is perpendicular to the direction of the wavelet, considered as the wavefronts.

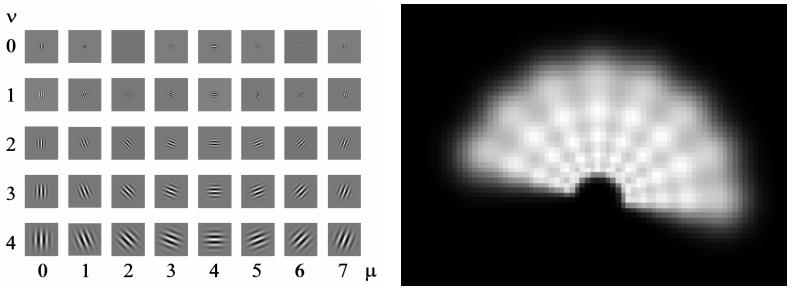


Fig. 1. Left: Bank of 40 Gabor filters, ordered by frequency (ν) and orientation (μ). Right: Response of Gabor filters in the Fourier space (only drawn half space).

The usual strategy for Gabor-based methods consists in convolving the images with the set of 40 filters and then working with the absolute value of the results [8]. As the dimensionality grows by a factor of 40 with these methods, many researchers have tackled this problem by combining Gabor wavelets with a dimension reduction algorithm, such as PCA. Analytic methods consider the Gabor responses computed only over a set of fiducial points, such as eyes, nose and mouth [9]. We shall call these methods "Feature-based Gabor PCA" (FGPCA). Holistic methods take into account the Gabor responses from the whole face image. Due to the huge dimensionality of Gabor features, a downsampling process is usually performed to reduce the dimension by a certain factor (usually 16 or 64) [10, 11, 12, 13]. We shall call these methods "Downsampled Gabor PCA" (DGPCA).

The algorithm proposed here is holistic, but it uses no downsampling process, as all the wavelet convolutions are performed in parallel, i.e., in a multi-channel approach. A final fusion of the results will allow us to evaluate the performance of our method and compare it with others mentioned above.

We also want to explore which of the 40 Gabor wavelets is able to extract the most discriminant features for a face verification problem. Some experiments have been done to study the influence of spatial frequency and orientation of face features [1, 14, 15, 16]. These works suggest that low frequency information can help us distinguish a face from a "non-face", but it is the high frequency information which is needed to

tell whether two faces are different. The importance of facial bilateral symmetry as a key element to identify a "beauty face" [17, 18] and its influence in the ability to recognize a face [19, 20] has been also considered in the past.

The remainder of this paper is organized as follows. In Section 2, we describe the face databases used. In Section 3 we explain our algorithm, the Parallel Gabor PCA. The design of our experiments can be found in Section 4. The results and a discussion are in Section 5. Finally, the conclusions are to be found in Section 6.

Fig. 2. Sample images from XM2VTS Database (above) and FRAV2D Database (below)

2 Face Databases

2.1 XM2VTS Database

XM2VTS is a multi-modal face database (Figure 2, top) [21] from the University of Surrey, UK, which comprises 2D pictures (frontal and profile views), as well as 3D meshes for 295 people. For our experiments we selected 100 people randomly, each having four frontal pictures taken in three different sessions. The first and the second sessions had frontal diffuse illumination, while in the third one the lighting was lateral (in two images the light came from the left and in the others, it came from the right). We use the set of four images from the first session of every person in the gallery database to train our classifiers. The remaining images from the other sessions are used in the tests to verify the accuracy of our algorithm. A manual process is used to normalize the face images. The images are cropped to 128×128 and converted into grey scale, with the eyes occupying the same locations in all the pictures. Finally a histogram equalization is performed on the images to deal with changes of illumination.

2.2 FRAV2D Database

We have also used the public domain FRAV2D Database (Figure 2, bottom) [22], which is freely available to the scientific community for research purposes. It comprises 109 people, each with 32 images. It includes frontal images with diffuse and zenithal illumination, 15º and 30º head orientations, and images with occlusions. As with XM2VTS Database, the images are normalized to 128×128 manually and histogram equalization is applied on them.

3 Our Algorithm

We have developed the so-called Parallel Gabor methods (Figure 3) [23, 24]. The core of the algorithms is a PCA-based dimension reduction process. However, unlike a standard PCA, a set of Gabor wavelet convolutions are applied to the gallery database, in order to extract information of frequency and orientation in the images. Following [7], a set of 40 wavelets (8 orientations and 5 frequencies) are used, so that the overall dimensionality of the problem increases by a factor of 40. Unlike other methods that try to tackle this huge dimensionality by downsampling the feature vectors, the Parallel Gabor methods do not perform any downsampling process at all, but they consider the images convolved with the same wavelet in parallel and independently. After the Gabor-based PCA, a set of person-specific SVM classifiers [25] are trained with the PCA projection coefficients. In this scenario, the images of one person are considered as genuine cases and the remaining ones are impostors. As we work with the Gabor convolutions in parallel, there are 40 different SVMs per person, each corresponding to a wavelet frequency and orientation.

The same steps are applied for the images in the test database. In this case, the PCA projection matrix learnt in the previous step is applied to these images in order to compute the PCA coefficients. These will be fed into the SVM classifiers in order to obtain a set of 40 scores, each one for every wavelet frequency and orientation, which are averaged so as to produce a final score [24]. With the fused scores, an overall equal-error rate (EER), for which the false acceptance rate equals the false rejection rate, is computed in order to characterize the goodness of the method.

In this paper, we compare our algorithm with others such as a standard PCA, a FGPCA (with 14 features, 8 for the occluded images and the 30º-turned images) and a DGPCA with a downsample factor of 16. An alternative one-Gabor PCA method that performs no data fusion has been implemented, in order to find out the influence of the wavelet frequency and orientation on the final EER.

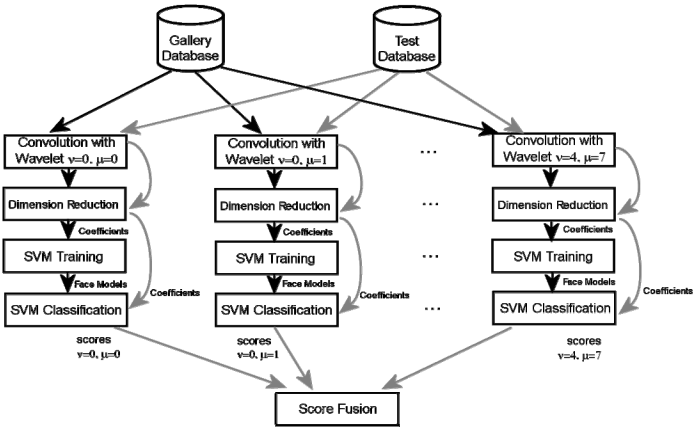


Fig. 3. Outline of our algorithm (PGPCA). Black arrows indicate the SVM training stage and grey arrows show the SVM test phase.

4 Design of Experiments

Six experiments were carried out (Table 1), for which a set of 4 frontal images per person was used to train the SVM classifiers. In every experiment a test with a disjoint set of 4 images per person was completed in order to compute the overall EER of the system, which has to be done by considering the scores for all the person-specific SVMs. Due to the configuration of the XM2VTS Database, only two tests could be performed, although six experiments are available for FRAV2D. As well there is a slight difference of meaning for test 6: While for FRAV2D the light direction changes from frontal diffuse to zenithal (which produces some shadows under the face features, such as the eyebrows, the nose and the mouth), for XM2VTS the illumination changes from frontal diffuse to lateral. This yields a dramatic effect on the images (half face is lit, while the other part is in shadow) and should be taken into account when comparing the results of this experiment for both databases.

Table 1. Specification of our experiments

Experiment	Images per person in gallery set	Images per person in test set	FRAV2D Database	XM2VTS Database
1	4 (neutral expression)	4 (neutral expression)	✓	✓
2		4 (15º turn)	✓	
3		4 (30º turn)	✓	
4		4 (gestures)	✓	
5		4 (occlusions)	✓	
6		4 (illumination)	✓	✓

5 Results and Discussion

5.1 Performance of Parallel Gabor PCA Versus Other Methods

In Figures 4 and 5 we present the EER with respect to the dimensionality, that is, the number of PCA coefficients kept after the dimension reduction, using different Gabor-based methods (FGPCA, DGPCA and PGPCA). A standard PCA has also been included as a reference.

Figure 4 shows the results for the XM2VTS Database. When few eigenvalues (60 – 70) are kept, PGPCA always obtains the lowest error compared to the other methods. This means that PGPCA succeeds even when an important dimension reduction is performed in the PCA stage. However, if we consider a higher dimensionality, DGPCA seems to obtain the lowest EER (1.0%), just slightly better than PGPCA (1.2%), in test 1 (Figure 4, left). In test 6 (Figure 4, right), DGPCA outperforms clearly the other methods with an EER 8.5% (PGPCA can only achieve 12.3%).

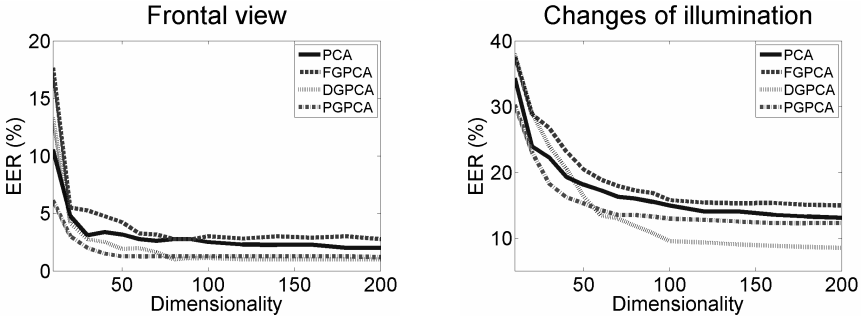


Fig. 4. Evolution of the EER as a function of the dimensionality (number of coefficients kept in the dimension reduction) for XM2TVS Database. From left to right: tests 1 and 6.

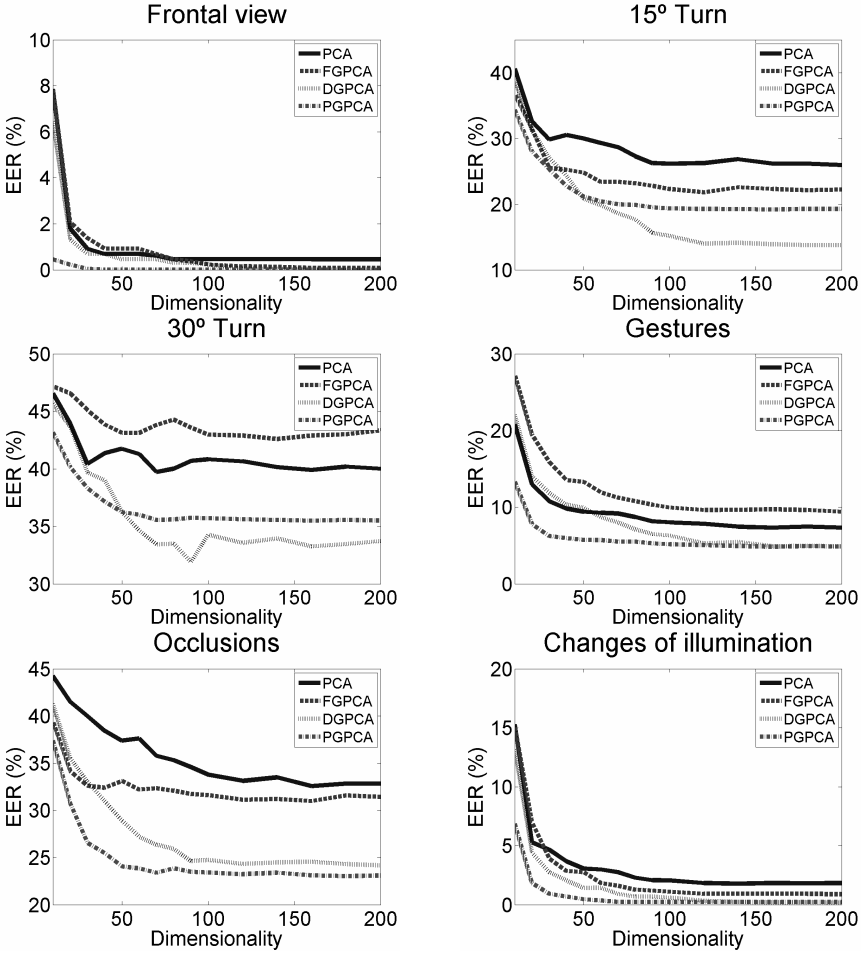


Fig. 5. Evolution of the EER as a function of the dimensionality for FRAV2D Database. From left to right, top to bottom: tests 1 to 6.

In Figure 5 we present the results for the FRAV2D Database. In this case, PGPCA obtains the lowest EER for test 1 (0.00%), test 4 (4.88%) and test 5 (23.04%), beating DGPCA (0.01%, 4.89% and 24.17%, respectively). For test 6, both methods obtain similar results (DGPCA 0.17%, PGPCA 0.23%). However, tests 2 and 3 (turns) show that DGPCA outperforms easily the other methods, included PGPCA (13.76% vs. 19.26% for a 15º head orientation, and 33.26% vs. 35.52% for a 30º head orientation). Therefore, for the FRAV2D Database, PGPCA methods achieves clearly the lowest error in three out of six experiments (tests 1, 4 and 5), although it obtains a slightly worse EER with respect to DGPCA in test 6. On the contrary, DGPCA outperforms PGPCA for tests 2 and 3. The other baseline methods, a standard PCA and FGPCA, always obtain the worst EERs for both databases in all experiments and have been included here only to help the comparison of results.

5.2 Influence of Gabor Wavelet Frequency and Orientation in a One-Gabor PCA

We have carried out another experiment in order to investigate the discriminant capabilities of Gabor wavelets. In this case, all the images in the gallery database are convolved with a unique Gabor wavelet of a certain frequency ν and orientation μ. With no downsampling and after a PCA dimension reduction process, the feature vectors are used to train a set of person-specific SVMs, just like in the previous section. However, the main difference here is that no score fusion is performed. Therefore, we have obtained a set of 40 EERs, each one for every Gabor wavelet, repeated for the six experiments in Table 1. The goal of this section is to learn which wavelet, when considered alone, is able to extract the face features with the highest distinguishing properties.

Figure 6 plots the EER as a function of the wavelet frequency (ν) for all orientations (μ) for both databases. For simplicity only the results for test 1 are shown (the corresponding figures for the other experiments are similar). This figure shows that, as a general rule, the wavelets with a higher frequency (low ν) give a better EER than the wavelets with a lower frequency (high ν), for both databases and all the experiments. This can be easily understood, as the low frequency information allows distinguishing a face from a "non-face", but it is not enough to separate two similar faces. It is the high frequency information which provides the necessary details to tell one face from the other.

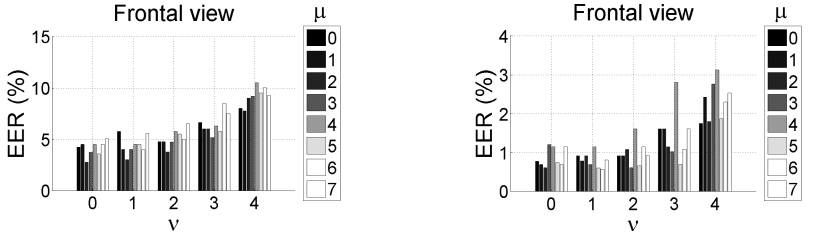


Fig. 6. Evolution of the EER as a function of the wavelet frequency (ν) for all orientations (μ) for XM2VTS (left) and FRAV2D (right) for test 1

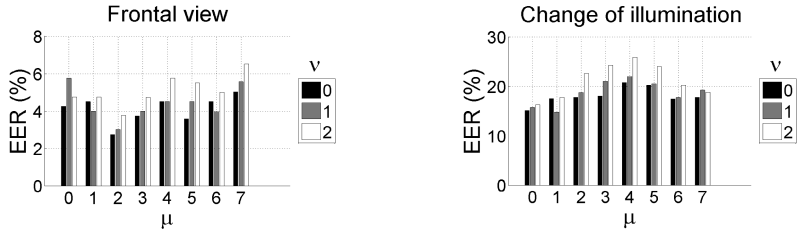


Fig. 7. Evolution of the EER as a function of the wavelet orientation (μ) for all frequencies (ν) for XM2VTS Database (left: test 1, right: test 6)

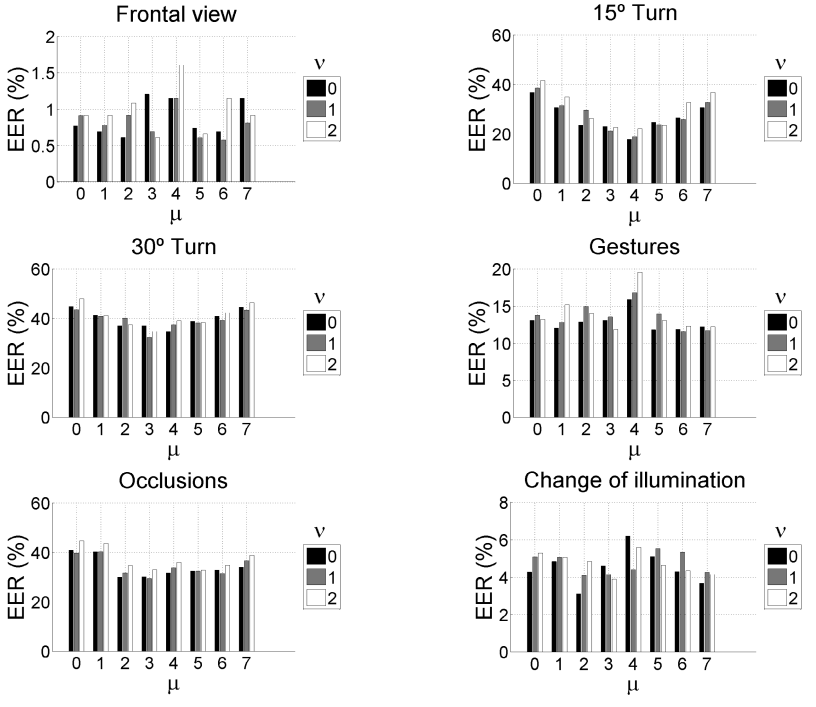


Fig. 8. Evolution of the EER as a function of the wavelet orientation (μ) for all frequencies (ν) for FRAV2D Database (left to right, top to bottom: test 1 to 6)

Figures 7 and 8 show the influence of the wavelet orientation μ. Only the high frequency wavelets have been considered ($0 \leq \nu \leq 2$), as we have seen they are more discriminant. For the XM2VTS Database, the wavelet with the lowest EER is achieved with orientation $\mu=2$ for test 1. Despite the face features can be horizontal (eyebrows, eyes, nostrils, mouth) or vertical (nose), the most influential wavelet extracts information from the lower left to the upper right corner of the image. On the contrary, in test 6, the best wavelet is the one with $\mu=0$, which clearly coincides with the illumination gradient direction. With respect to FRAV2D Database, except for test

2 (with 15º head orientation), oblique wavelets (μ=2, 3, 5, 6) usually have more discriminant power compared to horizontal (μ=4) or vertical wavelets (μ=0).

Another interesting conclusion is that the distribution of the EER as a function of μ is not symmetrical with respect to the central wavelet μ=4, despite the symmetry of a pair of wavelets with parameters μ and 8–μ. The exception is test 2 (images with 15º head orientation). Bearing in mind that the images in the database have been corrected from tilt, this can be understood as evidence that faces are not perfectly symmetrical. Our results seem to agree with those of [19, 20], which state that asymmetrical faces are easier to recognize than their symmetrical counterparts. Specifically, we have seen that some wavelet orientations produce a lower EER compared to the corresponding symmetrical ones, which means that in some cases the left half of the face carries more discriminant information that the right half, or vice versa.

6 Conclusions

We have presented the results of a thorough study of the so-called Parallel Gabor PCA algorithm for XM2VTS and FRAV2D Databases. Our algorithm outperforms other methods, such as PCA, FGPCA and DGPCA, when fewer PCA coefficients are kept. It has also obtained the best EER in three out of six experiments. When it ranked second, the final EER was only slightly worse compared to DGPCA. However, for images with significant head orientation, DGPCA is clearly the most effective.

In a one-Gabor PCA scenario we have seen that the features extracted by the high frequency ($0 \le v \le 2$) and oblique orientations (45º – 135º) wavelets are the most discriminant, as they have achieved the lowest EER. The different performance of wavelets and their mirrored equivalents shows that faces are not perfectly symmetrical and that those asymmetries carry more discriminant information. The experiments performed with the images with lateral lighting also show that the optimal wavelet is the one with a wave vector oriented in the illumination direction.

Acknowledgments. Part of this work has been supported by the Universidad Rey Juan Carlos under the financial program of mobility for teaching staff. Special thanks have also to be given to Professor Ian Dryden from the School of Mathematical Sciences of the University of Nottingham, UK, for his interesting comments.

References

1. Chellappa, R., Wilson, C.L., Sirohey, S.: Human and Machine Recognition of Faces: A Survey. Proceedings of the IEEE 83(5), 705–740 (1995)
2. Zhao, W., Chellappa, R., Phillips, J., Rosenfeld, A.: Face Recognition: A Literature Survey. ACM Computing Surveys, 399–458 (2003)
3. Turk, M., Pentland, A.: Eigenfaces for Recognition. Journal of Cognitive Neuroscience 3(1), 71–86 (1991)
4. Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J.: Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. IEEE PAMI 19(7), 711–720 (1997)
5. Daugman, J.G.: Uncertainty relation for resolution in space, spatial-frequency and orientation optimized by two-dimensional visual cortical filters. Journal of the Optical Society of America A: Optics Image Science and Vision 2(7), 1160–1169 (1985)

6. Lades, M., Vorbrüggen, J.C., Buhmann, J., Lange, J., von der Malsburg, C., Würtz, R.P., Konen, W.: Distortion invariant object recognition in the Dynamic Link Architecture. IEEE Transactions on Computers 42(3), 300–311 (1993)
7. Wiskott, L., Fellous, J.M., Kruger, N., von der Malsburg, C.: Face recognition by Elastic Bunch Graph Matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7), 775–779 (1997)
8. Shen, L., Bai, L.: A review on Gabor wavelets for face recognition. Pattern Analysis & Applications 9(2-3), 273–292 (2006)
9. Chung, K.-C., Kee, S.C., Kim, S.R.: Face Recognition using Principal Component Analysis of Gabor Filter Responses. In: International Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time Systems, pp. 53–57 (1999)
10. Liu, C.J., Wechsler, H.: Gabor feature based classification using the enhanced Fisher linear discriminant model for face recognition. IEEE Transactions on Image Processing 11(4), 467–476 (2002)
11. Shen, L., Bai, L.: Face recognition based on Gabor features using kernel methods. In: 6th IEEE Conference on Face and Gesture Recognition, pp. 170–175 (2004)
12. Gabor wavelets and general discriminant analysis for face identification and verification. Journal of Image and Vision Computing 27, 1758–1767 (2006)
13. Qin, J., He, Z.-S.: A SVM face recognition method based on Gabor-featured key points. In: Yeung, D.S., Liu, Z.-Q., Wang, X.-Z., Yan, H. (eds.) ICMLC 2005. LNCS (LNAI), vol. 3930, pp. 5144–5149. Springer, Heidelberg (2006)
14. Gilbert, C., Bakan, P.: Visual Asymmetry in Perception of Faces. Neuropsychologia 11(3), 355–362 (1973)
15. Rhodes, G.: Perceptual Asymmetries in Face Recognition. Brain and Cognition 4(2), 197–218 (1985)
16. Mitra, S., Lazar, N.A., Liu, Y.: Understanding the role of facial asymmetry in human face identification. Journal Statistics and Computing 17(1), 57–70 (2007)
17. Burt, D.M., Perrett, D.I.: Perceptual asymmetries in judgements of facial attractiveness, age, gender, speech and expression. Neuropsychologia 35(5), 685–693 (1997)
18. Fink, B., Neave, N., Manning, J.T., Grammer, K.: Facial symmetry and judgements of attractiveness, health and personality. Personality And Individual Differences 41(3), 491–499 (2006)
19. Tjan, B.S., Liu, Z.L.: Symmetry impedes symmetry discrimination. Journal of Vision 5(10), 888–900 (2005)
20. Brady, N., Campbell, M., Flaherty, M.: Perceptual asymmetries are preserved in memory for highly familiar faces of self and friend. Brain and Cognition 58(3), 334–342 (2005)
21. Messer, K., Matas, J., Kittler, J., Luettin, J., Maitre, G.: XM2VTS: The Extended M2VTS Database. In: 2nd International Conference on Audio and Video-based Biometric Person Authentication, pp. 72–77 (1999)
22. FRAV2D Database (2004), Freely available from: http://www.frav.es/databases/frav2d/
23. Serrano, Å., Conde, C., de Diego, I.M., Cabello, E., Bai, L., Shen, L.: Parallel Gabor PCA with Fusion of SVM Scores for Face Verification. In: International Conference on Computer Vision Theory and Applications, pp. 149–154 (2007)
24. Serrano, Å., de Diego, I.M., Conde, C., Cabello, E., Bai, L., Shen, L.: Fusion of Support Vector Classifiers for Parallel Gabor Methods Applied to Face Verification. In: Haindl, Kittler, J., Roli, F. (eds.) MCS 2007. LNCS, vol. 4472, pp. 141–150. Springer, Heidelberg (2007)
25. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)

Wrapping the Naive Bayes Classifier to Relax the Effect of Dependences

Jose Carlos Cortizo[1,2], Ignacio Giraldez[2], and Mari Cruz Gaya[2]

[1] Artificial Intelligence & Network Solutions S.L.
jccp@ainetsolutions.com http://www.ainetsolutions.com/jccp
[2] Universidad Europea de Madrid
Villaviciosa de Odon, 28670, Madrid, Spain
{josecarlos.cortizo, ignacio.giraldez, mcruz}@uem.es

Abstract. The Naive Bayes Classifier is based on the (unrealistic) assumption of independence among the values of the attributes given the class value. Consequently, its effectiveness may decrease in the presence of interdependent attributes. In spite of this, in recent years, Naive Bayes classifier is worked for a privilege position due to several reasons [1]. We present DGW (Dependency Guided Wrapper), a wrapper that uses information about dependences to transform the data representation to improve the Naive Bayes classification. This paper presents experiments comparing the performance and execution time of 12 DGW variations against 12 previous approaches, as constructive induction of cartesian product attributes, and wrappers that perform a search for optimal subsets of attributes.

Experimental results show that DGW generates a new data representation that allows the Naive Bayes to obtain better accuracy more times than any other wrapper tested. DGW variations also obtain the best possible accuracy more often than the state of the art wrappers while often spending less time in the attribute subset search process.

Keywords: DGW, Naive Bayes, (In)Dependence Assumption, Wrapper, Feature Evaluation and Selection.

1 Introduction and Motivation

There exist many approaches to the classification problem, from induction of decision tress, nearest neighbours approaches, etc. but the statistical approach seems to be the most intuitive an simple, in fact, there existed statistical approaches to classification previous than the machine learning ones [2]. From statistical classificators, Naive Bayes, which is based on the Bayes Theorem [3] is worked for a privilege position [1] due to its simplicity, its resilience to noise, its time and space efficiency [4], [5], its undernstandability [6], its results both in performance and speed in the area of information retrieval and automated text categorization [7], [8] and also in other areas and because it is well known that when independence assumption is held, no other classifier can outperform Naive Bayes in the sense of misclassification probability [9]. [10] shows the Naive Bayes

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 229–239, 2007.

classifier is competitive with other learning algorithms as decision trees and neural networks and, in certain situations, outperforms them. [11] and [12] present good experimental results for the Naive Bayes when compared versus other more modern machine learning algorithms. [13] shows that the Naive Bayes outperforms the most state-of-the-art decision-tree based algorithms for ranking.

The Naive Bayes classifier is based upon the simplifying assumption of conditionally independent attributes given the class value (see [14] and [15] for more detailed explanations). This assumption is not very realistic as in many real situations the attributes are, in some manner, dependent.

There are many attempts to relax this assumption in the literature, which can be summarized in three main trends:

1. Attempts that try to relax this assumption by modifying the classifier [16], [17].
2. Feature extraction in order to modify the input data to achieve independent (or pseudo-independent) attributes [4], [18], [15].
3. Approaches that underestimate the independence assumption [19], [5], [20], [21].

This article, that presents an approach in the line of 2, is organized as follows. In the next section general issues related to wrapper methods and its applications to the Naive Bayes classifier and the independence assumption are presented. In Section 3 the Dependence Guided Wrapper (DGW), a correlation guided wrapper that performs an attribute subset selection process for the Naive Bayes is presented. Section 4 explains the experiments made for testing the DGW algorithm and the comparisons to other attribute selection methods. Section 5 discusses the results and the last section presents some conclusions and future work.

2 Wrappers for Attribute Subset Selection

The prediction accuracy of classifiers may degrade in prediction accuracy when faced with many irrelevant and/or redundant attributes. The explanation to this phenomenon may be found in the "Curse of the Dimensionality" [22] which refers to the exponential growth of the number of instances needed to describe the data as a function of dimensionality (number of attributes). $\phi(\mathbf{X}, \mathbf{A})$ is a function that transforms a dataset $\mathbf{X}$ to contain only the attributes included in $\mathbf{A}$. Given a dataset $\mathbf{X}$ containing a set of features $\mathbf{A}$ and given a certain learning algorithm L, Attribute Subset Selection tries to achieve a subset of the original attributes $A_F \subset \mathbf{A}$ such that when running L using $\phi(\mathbf{X}, A_F)$ as data to learn, L obtains the highest possible accuracy. There are two main trends in Attribute Subset Selection: the **filter model** and the **wrapper model**.

While the **filter model** [23] is based upon the idea of Relevance [24] and selects the attributes independently of what classifier will be used, the **wrapper model** [25] conducts a search in the space of possible attributes using the performance of the classifier as the evaluation function (or at least as an important

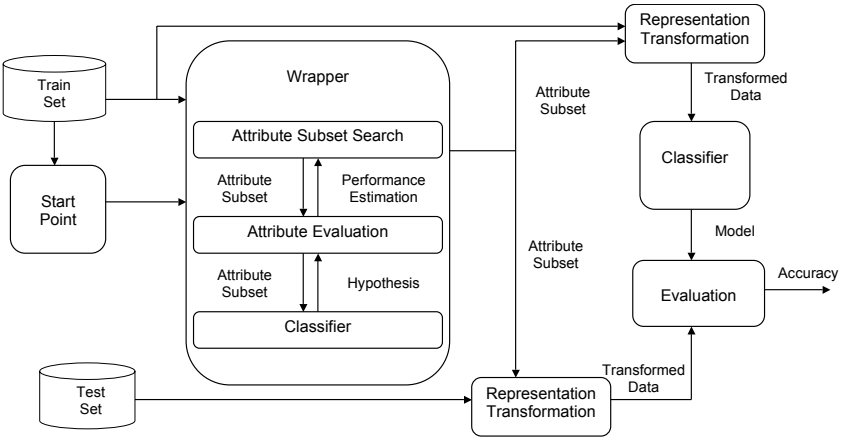


Fig. 1. General structure of a Wrapper

part of the evaluation function). Figure 1 shows the general structure of a wrapper where the initial training set is given to the wrapper with an initial starting point of attributes (all the attributes, none of the attributes or a random set of attributes) and then the wrapper conducts a search (according to the attribute selection search selected) where the induction algorithm is used as a black box to measure the performance of each evaluated subset. Finally, the initial data representation is transformed to complain the final attribute subset achieved.

[26] studied a wrapper performing a forward greedy search using the Naive Bayes as induction algorithm (FSS, Forward Sequential Selection). FSS tries to deal with highly correlated attributes by incorporating only some attributes in the final attributes set. [27] compared FSS and an adaption of Kittler's work [28] called BSE (Backward Sequential Elimination) very similar to FSS but performing a backwards greedy search. [18] proposed joining as an operation that creates a new compound attribute that replaces the original two dependent attributes, then explored two alternative methods related to FSS and BSE: FSSJ (Forward Sequential Selection and Joining) and BSEJ (Backward Sequential Elimination and Joining). Both, FSSJ and BSEJ, maintain a set of attributes to be used by the classifier and operates in a similar way: at each step the algorithm considers a set of possible operations (eliminate/select one attribute or joining two attributes) and studies the effect of each operation in the accuracy obtained by the classifier. The change that makes the most improvement is retained and the process is repeated until no accuracy improvement is achieved.

Experimental results show that the inductive construction of attributes helps the wrapper to achieve greater accuracies. FSS, BSE, FSSJ and BSEJ improve the Naive Bayes performance but due to the structure of a wrapper (uses the accuracy of the classifier as a metric of each possible subset of attributes) other wrappers might also improve the Naive Bayes performance.

3 The Dependency Guided Wrapper

This paper presents a wrapper attribute selection method that iteratively discards the top-N linearly correlated features (with N equal to 1, 2 and 3 in experiments) until the accuracy of the Naive Bayes classifier does not improve. Then it transforms the original data set so it complies with the new data representation. This method is called Dependency Guided Wrapper (DGW) as it tries to find a representation free of dependent attributes to satisfy the independence assumption. Figure 1 shows the general structure of a Wrapper. The DGW Wrapper is presented by describing each of the components presented in the general structure.

Let $\mathbf{A} = \{A_1, A_2, \dots A_n\}$ be the original set of attributes and $\mathbf{Y} = \{y_1, y_2, \dots y_m\}$ the possible class values, $\mathbf{x}_k = \{\boldsymbol{x}, y\}$ is a training example where $\boldsymbol{x} = (a_1, a_2, \dots a_n)$ is a point that belongs to the input space $\mathbf{X}$ and $y \in \mathbf{Y}$ is a point belonging to the output space $\mathbf{Y}$.

- ▷ **Starting Point:** The DGW Wrapper starting point contains all the original attributes.
- ▷ **Classifier:** The classifier used to estimate the performance of each attribute subset is the Naive Bayes Classifier
- ▷ **Attribute Selection Search:** We perform a two stage search.
  - The first stage is the **Dependency Based Wrapping**. The DGW tries to avoid all the dependences present on the original attributes. For that, DGW measures the Coefficient of Determination (R^2) of the linear dependence between each pair of attributes as a way to estimate how dependent they are. Each dependence, d_i is defined as the two related attributes and the R^2 values for those attributes, $d_i = (A_i, A_j, R^2(A_i, A_j))$. Then, it constructs a list, L_{d_i}, containing each possible dependence d_i, in a strength-of-dependency decremental order. Then, L_{d_i} is used to obtain the final attributes by deleting the most dependent attributes until no accuracy improvement is achieved.

    At each step, the algorithm considers a given number of dependences (N), the first N in L_{d_i}. As each dependence is represented by two attributes A_i and A_j, at each iteration, DGW deals, at most, with $2 * N$ attributes. At each iteration, for each of the $2 * N$ attributes, DGW calculates the impact in the classifier's accuracy of removing the attribute from the attributes set. At the end of each iteration, DGW either deletes the attribute which its impact when removing produces a higher increase on the final accuracy, or stop the Attribute Selection Search when all the final accuracies imply a fall in the accuracy of the classifier.
  - The second stage is the **Attribute Evaluator Ranker**. For each attribute not deleted in the previous stage, DGW calculates its value according to a certain Attribute Evaluator and then generate a list L_{AE} containing all these attributes ordered in this metric decremental order. For each attribute on the list, the impact on the final accuracy when

deleting that attribute is studied. If DGW obtains a higher or the same accuracy when deleting it, DGW then removes the attribute from the list.

Once finished the whole process, DGW obtains $A_F = \{A_i, A_j, \ldots A_z | A_k \in A\}$ that is a subset of the original attributes that contains the best ones.

4 Experiments

In this section some experiments to evaluate the proposed wrapper are presented. These experiments compare DGW to other methods and wrappers that perform a search among the possible subsets of attributes to improve the performance of the Naive Bayes Classifier. For this purpose, 23 datasets have been selected from the University of California at Irvine Machine Learning Repository [29] (see Table 1) trying to show a certain variety in the number of attributes (from 4 to 58) and instances (from a few more than 100 to almost 50.000).

We have selected the wrappers compared in Pazzani's work [27] [18]: Forward Sequential Selection (FSS), Backward Sequential Elimination (BSE), Forward Sequential Selection and Joining (FSSJ), Backward Sequential Elimination and Joining (BSEJ) but also some other wrappers from the state of the art of wrappers included in Weka [30] (using their default parameters): Best First Forward (BFF), Best First Backward (BFB), Best First Bidirectional (BFBi), Genetic Search (GS), Ranker using Information Gain (IG), Ranking with Gain Ratio (GR), Ranking with Relief (RE), Ranking with Squared-Chi (C2).

DGW has been tried using 1, 2 or 3 dependences at each step and using four different attribute evaluators (Information Gain, Relief, OneR and Symmetrical Uncertainty), which means 12 possible combinations: Information Gain and 1 (1IG), 2 (2IG) or 3 (3IG) dependences studied at each step, Relief combined with 1 (1RE), 2 (2RE) or 3 (3RE) dependences at each step, OneR with 1 (1OR), 2 (2OR) or 3 (3OR) dependences and Symmetrical Uncertainty combined with 1 (1SU), 2 (2SU) or 3 (3SU) dependences at each step.

In order to evaluate all the studied methods, we have performed a 10 times 10-fold cross validation obtaining the average accuracy and the standard deviation for each wrapper and each dataset. We have also performed a two-tailed t-test at the .05 level to determine whether each wrapper has a significant effect on the accuracy of the Naive Bayes classifier. Table 2 resumes the results, showing for each algorithm and for each dataset whether the preprocessing algorithm obtains a significantly better data representation (marked as +), worse (−) or if the preprocessing algorithm does not achieve a significantly different data representation (blank). N means the algorithm is too slow and there is no results for the dataset (some algorithms are hundreds of times slower than the Naive Bayes).

At the end of Table 2 there is a summary showing the number of times each algorithm achieves a significantly better data representation (Total-23). As for some algorithms there are only results for 15 datasets, there is also a summary (Total-15) counting the number of times each algorithm achieves a better data

Table 1. Descriptions of the datasets used. #att means the number of attributes of the dataset, #inst the number of instances, #nom the number of nominal attributes, #num the number of numeric attributes, NB Acc. the average accuracy of the Naive Bayes classifier and NB Discrete the average accuracy of the Naive Bayes classifier when discretizing all the attributes. The numbers after the ± indicate the standard deviation of the reported accuracy. All the results are obtained by 10 times 10-fold cross validation.

Dataset	#atts.	#inst.	#nom.	#num.	NB Acc	NB Discrete
Haberman	4	306	1	3	$74.80^{\pm 0.2}$	$74.38^{\pm 0.5}$
Hayes-Roth	5	132	1	4	$72.18^{\pm 4.0}$	$81.13^{\pm 1.3}$
Iris	5	150	1	4	$95.46^{\pm 0.6}$	$93.06^{\pm 0.7}$
Tae	6	151	5	1	$51.56^{\pm 1.2}$	$52.25^{\pm 1.3}$
Bupa	7	345	1	6	$55.14^{\pm 0.8}$	$57.39^{\pm 1.7}$
Yeast	7	1.479	1	6	$57.32^{\pm 0.4}$	$50.92^{\pm 0.5}$
Machine	8	209	1	7	$66.80^{\pm 2.4}$	$44.11^{\pm 0.7}$
Ecoli	8	336	1	7	$85.61^{\pm 0.7}$	$84.09^{\pm 0.5}$
Pima Indians D.	9	768	1	8	$75.75^{\pm 0.5}$	$75.01^{\pm 0.2}$
Abalone	9	4.177	2	7	$23.99^{\pm 0.1}$	$24.27^{\pm 0.1}$
Nursery	9	12.960	9	0	$90.28^{\pm 0.0}$	$90.28^{\pm 0.0}$
Glass	10	214	1	9	$48.14^{\pm 1.3}$	$50.88^{\pm 1.7}$
TicTacToe	10	958	10	0	$69.81^{\pm 0.3}$	$69.81^{\pm 0.3}$
Cmc	10	1.473	8	2	$50.63^{\pm 0.3}$	$48.53^{\pm 0.4}$
Wine	14	178	1	13	$97.46^{\pm 0.4}$	$97.12^{\pm 0.5}$
Adult	15	48.842	9	6	$82.69^{\pm 0.0}$	$81.54^{\pm 0.0}$
Crx	16	653	10	6	$77.75^{\pm 0.3}$	$86.02^{\pm 0.3}$
PenDigits	17	10.992	1	16	$85.77^{\pm 0.1}$	$85.59^{\pm 0.1}$
Letter Recog.	17	20.000	1	16	$64.07^{\pm 0.1}$	$59.98^{\pm 0.1}$
Segmentation	20	2.310	1	19	$80.10^{\pm 0.1}$	$89.58^{\pm 0.1}$
WdbCancer	31	569	1	30	$93.32^{\pm 0.2}$	$94.06^{\pm 0.2}$
Ionosphere	34	351	1	33	$82.48^{\pm 0.6}$	$88.92^{\pm 0.3}$
Spam	58	4.610	1	57	$79.51^{\pm 0.1}$	$75.41^{\pm 0.1}$

representation only taking care about the 15 datasets for which all algorithms have been computed.

For allow comparisons, Figure 2 and Figure 3 show the number of times each wrapper achieves the best accuracy. Figure 2 groups the state of the art wrappers and Figure 3 shows the results for the DGW variations. Dark bars represents the number of times each wrapper achieves the best accuracy when only the 15 datasets for which all wrappers have results, and light bars show the results when taking into account all the 23 datasets.

We have also computed the execution times of the wrappers ir order to allow a comparison among them (see Figures 4 and 5). For each dataset, we have computed the number of times each wrapper is slower than the Naive Bayes (as a way to avoid absolute times and normalize the data). Then, for each wrapper, we have computed the average of these values, resulting in a metric that shows the number of times each wrapper is slower than the Naive Bayes. Figure 4

Table 2. Summary table showing when a wrapper achieves a significantly better data representation (+) or worse (−) than the original one. *N* means there are no results available for that wrapper and dataset. A blank space means the data representation achieved by the wrapper is not significantly distinct than the original one. All the variations of the DGW algorithm are condensed in DGW(All) as all of them achieve significantly better data representations in the same datasets.

Data Set	FSS	BSE	FSSJ	BSEJ	BFF	BFB	BFBi	GS	GR	IG	RE	C2	DGW(All)
Tae			+	+									
Hayes-R.			+	+									+
Haber.													
Iris													
Bupa	+	+	+	+	+	+	+	+	+	+	+	+	+
Wine		+		+	+	+	+	+					+
Machine	+	+	−	−	+	+	+	+	+	+	+	+	+
Glass	+	+	+	+	+	+	+	+	+	+			+
Ecoli			−										
Pima Ind.	+	+	+	+	+	+	+	+					+
TicTacT.		+	+	+	+	+	+	+	+	+	+	+	+
Crx	*N*	*N*	*N*	*N*	+	+	+	+	+	+	+	+	+
Cmc	+	+			+	+	+	+	+	+		+	+
Yeast			−	−									
Ionosp.	*N*	*N*	*N*	*N*	+	+	+	+	+	+	+	+	+
WdbC.	*N*	*N*	*N*	*N*	+	+	+	+	+	+	+	+	+
Abalone	+	+	+	+	+	+	+	+	+	+	+	+	+
Segmen.	*N*	*N*	*N*	*N*	+	+	+	+	+	+	+	+	+
Spam	*N*	*N*	*N*	*N*	+	+	+	+	+	+	+	+	+
Letter R.	*N*	*N*	*N*	*N*	+	+	+	+	+	+	+	+	+
Pendigits	*N*	*N*	*N*	*N*	+	+	+	+					+
Nursery	−		+	+									
Adult	*N*	*N*	*N*	*N*	−	+	−	+		+	+	+	+
Total-15	5	8	5	7	8	8	8	8	6	6	4	5	9
Total-23	*N*	*N*	*N*	*N*	14	16	14	16	12	13	11	11	17

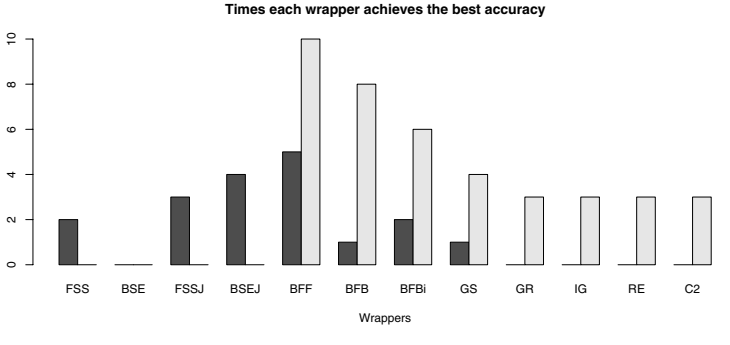


Fig. 2. Number of times each wrapper from the state of the art achieves the best accuracy. Dark bars represents the results for 15 datasets and light bars show the results for all the 23 datasets.

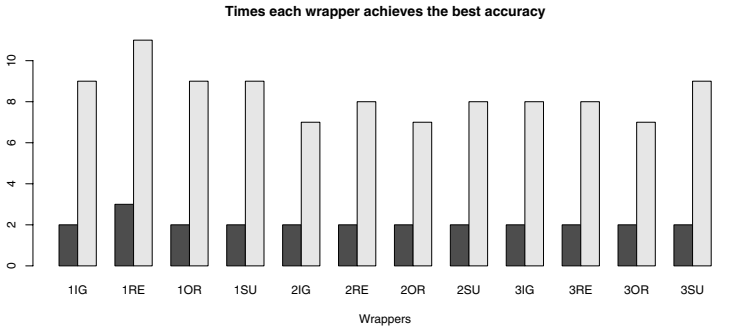


Fig. 3. Number of times each DGW variation achieves the best accuracy. Dark bars represents the results for 15 datasets and light bars show the results for all the 23 datasets.

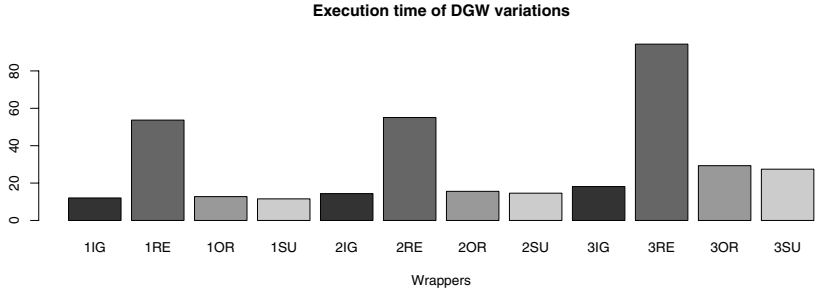


Fig. 4. Execution times (computed as the number of times each algorithm is slower than the Naive Bayes) of the DGW variations

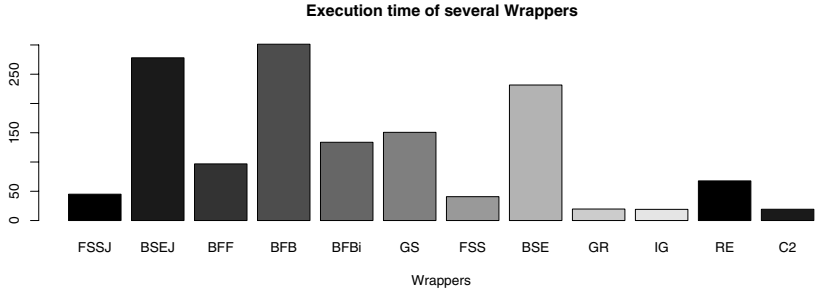


Fig. 5. Execution times (computed as the number of times each algorithm is slower than the Naive Bayes) for the state of the art wrappers compared in this paper

groups the results for the DGW variations and Figure 5 shows the results for the state of the art wrappers tested.

5 Discussion of Results

Comparing results presented in Table 2, it can be concluded that DGWrapper achieves significantly better data representations for the Naive Bayes more often than any other wrapper tested in the experiments: 17 times when considering all the datasets and 9 times when considering only 15. DGW (1R variation) obtains the best accuracy in 11 of the 23 datasets proven, more often than the rest of the wrappers. BFF obtains the best accuracy 10 times, not very far from DGW results, but DGW is 40% faster than BFF. Using these experimental results we can conclude that DGW is the best wrapper when using the Naive Bayes Classifier as learner, in terms of final predictive accuracy. Moreover, in the execution time graphics, we can notice that the DGW is among the fastest wrappers, more than 100 times faster than the FSSJ and BSEJ proposed by Pazzani. The DGW is equiparable, in terms of running time, to the most simple and faster wrappers: the rankers.

Backward approaches can not be regarded as superior to forward approaches since although backward approaches (BSE, BSEJ, BFB) obtain significantly better data representations for the Naive Bayes more often that their corresponding forward approaches (FSS, FSSJ, BFF) (see Table 2), results also show that forward approaches obtain the best accuracy more often than the backward ones (see Figure 2).

FSSJ and BSEJ, in average, are as good as FSS and BSE, but in some datasets are quite better or worse than the rest of wrappers. FSSJ and BSEJ need all the attributes to be discrete, and it can be noticed in Table 1 that for those datasets, when discretizing all the attributes, the Naive Bayes Classifier shows a similar behavior. This means that part of the effect of the FSSJ and BSEJ is due to dicretization of the attributes. In any case the concept of joining attributes instead of deleting them seems very interesting and would be interesting to integrate in the proposed method allowing the DGWrapper to evaluate at each step whether to delete one of the attributes belonging to the dependence or to join the attributes.

6 Conclusions and Future Work

We have shown that when learning the Naive Bayes Classifier, searching for dependences among attributes results in significant increases in accuracy. We proposed a general algorithm, DGW, that performs a search in the attribute space guided by the information of linear dependences among the values of the attributes. We have tested 12 DGW variations and 12 state of the art wrappers, performing searches in the attribute space in order to avoid the effect of dependences in the Naive Bayes Classifier. Experimental results and evaluation show that DGW provides the most improvement while spending less time than almost any other wrapper tested.

BSEJ and FSSJ rely their performance in the discretization of the attributes, and that makes difficult to compare the results. In future experiments we would

test variations of BSEJ and FSSJ that do not need to discretize all the attributes (or that only discretizes the attributes 'on demand') in order to allow better comparison. Joining the attributes seems a good way to delete dependences but not loosing some other information and would be interesting to incorporate this concept into the DGW algorithm. Also would be interesting to test other metrics to measure the dependences among attributes like Mutual Information or multivariate measures like CFS [31].

References

1. Rish, I.: An empirical study of the naive bayes classifier. In: International Joint Conference on Artificial Intelligence, American Association for Artificial Intelligence, pp. 41–46 (2001)
2. Fisher, R.: The use of multiple measurements in taxonomic problems. Annals of Eugenics 7, 179–188 (1936)
3. Bayes, T.: An essay towards solving a problem in the doctrine of chances. Philosophical Transactions 53, 370–418 (1963)
4. Kononenko, I.: Semi-naive bayesian classifier. In: EWSL 1991. Proceedings of the European working session on learning on Machine learning, pp. 206–219. Springer, New York (1991)
5. Zhang, H., Ling, C.X., Zhao, Z.: The learnability of naive bayes. In: Hamilton, H.J. (ed.) AI 2000. LNCS (LNAI), vol. 1822, pp. 432–441. Springer, Heidelberg (2000)
6. Kononenko, I.: Inductive and bayesian learning in medical diagnosis. Applied Artificial Intelligence 7(4), 317–337 (1993)
7. Lewis, D.D.: Representation and learning in information retrieval. PhD thesis, Amherst, MA, USA (1992)
8. Lewis, D.D.: Naive (Bayes) at forty: The independence assumption in information retrieval. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 4–15. Springer, Heidelberg (1998)
9. Mitchell, T.: Machine Learning, 1st edn. McGraw Hill, New York (1997)
10. Michie, D., Spiegelhalter, D.J., Taylor, C.C.: Machine Learning, Neural and Statistical Classification. Ellis Horwood (1994)
11. Kononenko, I.: Comparison od inductive and naive bayesian learning approaches to automatic knowledge adquisition
12. Langley, P., Iba, W., Thompson, K.: An analysis of bayesian classifiers. In: National Conference on Artificial Intelligence, pp. 223–228 (1992)
13. Zhang, H., Su, J.: Naive bayesian classifiers for ranking. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 501–512. Springer, Heidelberg (2004)
14. Cortizo, J.C., Giráldez, J.I.: Discovering data dependencies in web content mining. In: Gutierrez, J.M., Martinez, J.J., Isaias, P. (eds.) IADIS International Conference WWW/Internet (2004)
15. Cortizo, J.C., Giráldez, J.I.: Multi criteria wrapper improvements to naive bayes learning. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 419–427. Springer, Heidelberg (2006)
16. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29(2-3), 131–163 (1997)

17. Kohavi, R.: Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 202–207 (1996)
18. Pazzani, M.: Constructive induction of cartesian product attributes. ISIS: Information Statistics and Induction in Science (1996)
19. Domingos, P., Pazzani, M.J.: On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning 29(2-3), 103–130 (1997)
20. Domingos, P., Pazzani, M.J.: Beyond independence: Conditions for the optimality of the simple bayesian classifier. In: International Conference on Machine Learning, pp. 105–112 (1996)
21. Hand, D.J., Yu, K.: Idiot's bayes - not so stupid after all? International Statistical Review 69(3), 299–385 (2001)
22. Bellman, R.: Adaptive Control Processes: a Guided Tour. Princeton University Press, Princeton (1961)
23. Duch, W.: Filter Methods. In: Feature Extraction, Foundations and Applications, Springer, Heidelberg (2004)
24. Langley, P.: Selection of relevant features in machine learning. In: Proceedings of the AAI Fall Symposium on Relevance (1994)
25. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
26. Langley, P., Sage, S.: Induction of selective bayesian classifiers, pp. 399–406 (1994)
27. Pazzani, M.J.: Searching for Dependencies in Bayesian Classifiers. In: 5thWorkshop on Artificial Intelligence and Statistics (1996)
28. Kittler, J.: Feature Selection and Extraction. In: Handbook of Pattern Recognition and Image Processing, Academic Press, London (1986)
29. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
30. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
31. Hall, M.A.: Correlation-based Feature Selection for Machine Learning. PhD thesis, Department of Computer Science, University of Waikato (1998)

Preference Learning from Interval Pairwise Data. A Distance-Based Approach

Esther Dopazo[1], Mauricio Ruiz-Tagle[2], and Juan Robles[1]

[1] Facultad de Informática, Technical University of Madrid, Campus de Montegancedo, CP28660, Boadilla del Monte (Madrid), Spain
{edopazo,jrobles}@fi.upm.es
[2] Facultad de Ciencias de la Ingeniería, Universidad Austral de Chile, General Lagos 2086, Campus Miraflores, Valdivia, Chile
mruiztag@uach.cl

Abstract. Preference learning has recently received a lot of attention within the machine learning field, concretely learning by pairwise comparisons is a well-established technique in this field. We focus on the problem of learning the overall preference weights of a set of alternatives from the (possibly conflicting) uncertain and imprecise information given by a group of experts into the form of interval pairwise comparison matrices. Because of the complexity of real world problems, incomplete information or knowledge and different patterns of the experts, interval data provide a flexible framework to account uncertainty and imprecision. In this context, we propose a two-stage method in a distance-based framework, where the impact of the data certainty degree is captured. First, it is obtained the group preference matrix that best reflects imprecise information given by the experts. Then, the crisp preference weights and the associated ranking of the alternatives are derived from the obtained group matrix. The proposed methodology is made operational by using an Interval Goal Programming formulation.

Keywords: Preference learning, pairwise comparison matrices, interval data, distance methods, interval goal programming.

1 Introduction

Preference learning has recently received a lot of attention within the machine learning literature [1] [2]. Learning by pairwise comparisons is a well-established technique in this field. In particular, it is a useful inference tool for assessing the relative importance of several alternatives. Formally, we consider the following preference learning scenario: let $X = \{x_1, ..., x_n\}(n \geq 2)$ be a finite set of alternatives (or labels) and $\{E_1, ..., E_m\}$ be a group of m experts. We assume that preference information provided by the expert E_k is structured by a pairwise comparison (pc) matrix $M^k = (m_{ij}^k)$, where $m_{ij}^k > 0$ represents the relative importance of the alternative i over the alternative j, given by expert k. In many applications, due to incomplete information or knowledge, unquantifiable information, imprecise data, etc., the information cannot be assessed precisely

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 240–247, 2007.

in a quantitative form, so interval assessments are a natural way for expressing preferences. Therefore, we will consider interval pc matrices in our work.

In this scenario, we focus on the problem of scoring and ranking alternatives by computing their crisp preference weights that best reflect interval pairwise preferences, $M^1, ..., M^m$, given by multiple experts.

In this context, we face with different problems: the imprecision management problem (expert preferences are expressed by interval data), the group problem (i.e., how to integrate preferences from multiple experts) and the problem of consistency (i.e., how to derive preference weights from interval pc matrices without consistency properties).

In the Analytic Hierarchy Process (AHP) [14] context, the problem of consistency for interval assessments is analysed in [3], [4], [5], [6] and [7]. On the same context, the consensus problem has been studied under the fuzzy approach in [8], [9] and [10]. In [11], a consistency-driven logarithmic goal programming approach is applied for dealing with interval data for a particular distance. A distance model for interval rankings has been proposed in [12].

The related works deal with imprecise data, but they do not consider data certainty degree in the learning process. Because in multiple experts problems information is non-homogeneous, it is represented by interval-valued data with different precision degree, we propose to consider it. On the other hand, most of the methods dealing with interval data lead to interval weights. When the interval weights overlap, there is no a unique ranking of alternatives. In this case, additional ranking procedures are required in order to compare the final alternative scores.

We propose a two-stage method in a general distance-based framework, where the impact of the data certainty degree is captured. First, a method to retrieve the group preferences from the conflicting and imprecise individual preferences is proposed. To do this, we look for the crisp information that best reflects the multiple experts preferences by using a l_p-metric relative to the precision data. In the second step, the overall preference weights of the alternatives are computing from the group preference information obtained in the first phase. The proposed approach is made operational with the help of Interval Goal Programming (GP).

The paper is organized as follows. Section 2 focuses on the formulation of the problem and describes the first stage of the proposed model, group preference learning. In section 3, it is presented the second stage of the model and finally, the main conclusions derived from this work are included in section 4.

2 Group Preference Learning

Let $X = \{x_1, ..., x_n\} (n \geq 2)$ be a finite set of alternatives and $\{E_1, ..., E_m\}$ a group of m experts. We assume that expert E_k is indecisive generating certain imprecision in his preferences. Consequently, he quantifies his preferences on the elements of X giving an interval pc matrix on X, $M^k = ([\underline{m_{ij}^k}, \overline{m_{ij}^k}])$ as follows: he judges that alternative i is between $\underline{m_{ij}^k}$ and $\overline{m_{ij}^k}$ times more important than

alternative j with $\underline{m_{ij}^k}$, $\overline{m_{ij}^k} > 0$ and $\underline{m_{ij}^k} < \overline{m_{ij}^k}$. Then, the interval comparison matrix M^k is obtained

$$M^k = \begin{pmatrix} 1 & \left[\underline{m_{12}^k}, \overline{m_{12}^k}\right] & \cdots & \left[\underline{m_{1n}^k}, \overline{m_{1n}^k}\right] \\ \left[\underline{m_{21}^k}, \overline{m_{21}^k}\right] & 1 & \cdots & \left[\underline{m_{2n}^k}, \overline{m_{2n}^k}\right] \\ \vdots & \vdots & \vdots & \vdots \\ \left[\underline{m_{n1}^k}, \overline{m_{n1}^k}\right] & \left[\underline{m_{n2}^k}, \overline{m_{n2}^k}\right] & \cdots & 1 \end{pmatrix} \tag{1}$$

In practice, the complexity of the problem, imperfect and subjective judgements, different backgrounds of the experts, etc., lead to imprecise and incompatible pairwise information. Also, disjoint intervals could be assigned by different experts to the same objects, i.e. there could exist i and j and two different experts such that $\left[\underline{m_{ij}^{k_1}}, \overline{m_{ij}^{k_1}}\right] \cap \left[\underline{m_{ij}^{k_2}}, \overline{m_{ij}^{k_2}}\right] = \emptyset$.

In this phase, the challenge is to look for the group preferences that represent in some sense the multiple experts preference acting as a whole. The ideal solution meaning unanimous agreement, among all experts regarding all possible alternatives is difficult to achieve in real-life situations. Therefore, we provide a method for retrieving the group preference information that best reflects the multiple experts preferences $(M^1, ..., M^m)$ attending to the data precision. We provide a method to obtain a matrix C, named group preference matrix, such that all experts consider the information of C to be close to their data. In order to measure the degree of closeness, we consider the l_p-distances family with weights relatives to data precision degree. Thus, we look for an $n \times n$ crisp positive matrix C, whose entries $c_{ii} = 1$ if $i = 1, .., n$ and c_{ij} is obtained for each pair (i, j) $(i \neq j)$ according to the following expression

$$\min_{c_{ij}>0} \left[\sum_{k=1}^{m} \left(\frac{\left| m_{ij}^k - c_{ij} \right|}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} \right)^p \right]^{1/p} \qquad if \ 1 \leq p < \infty \tag{2}$$

over the set of positive numbers.

For $p = \infty$ metric, for each (i, j) $(i \neq j)$, the problem turns into the *minmax* problem.

$$\min_{c_{ij}>0} \left\{ \max_{k=1,..,m} \left\{ \frac{\left| m_{ij}^k - c_{ij} \right|}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} \right\} \right\} \tag{3}$$

over the set of positive numbers.

We notice that in the above problems the input data are interval, $m_{ij}^k \in [\underline{m_{ij}^k}, \overline{m_{ij}^k}]$ that is the interval goal of the expert k for each entry (i, j).

The value $\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}$ is considered a measure of the imprecision degree of m_{ij}^k data given by expert k. Because we work with pairwise estimations of the

weight ratios, the logarithmic transformation is used to equalize the precision degree of the m_{ij}^k data and its reciprocal $1/m_{ij}^k$ (assuming interval arithmetic).

It should be noticed that as the range of the data (the vagueness) increases, less importance is given to this data in the objective function.

In the posed problems, the relative residual aggregation is affected by the parameter p of the distance. Thus as p increases, more importance is given to the largest relative residual value. The extremes of this set are the distance l_1, which minimizes the sum of relative residual and the Chebyshev or Minmax metric l_∞, which minimizes the maximum relative deviation. Metric $p = 2$ corresponds to the Euclidean distance, generating a least square problem.

Once the analytical model has been established, we focus on solving the proposed minimization problems. In order to board them with interval data, we consider Interval Goal Programming ([12] and [13]). In this context, for each pair (i, j) $(i \neq j)$, we consider the common deviational variables used in GP (see for example [12]):

$$\underline{n_{ij}^k} = \frac{1}{2}\left[\frac{\left|\underline{m_{ij}^k} - c_{ij}\right|}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} + \frac{\underline{m_{ij}^k} - c_{ij}}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}}\right] \tag{4}$$

$$\underline{p_{ij}^k} = \frac{1}{2}\left[\frac{\left|\underline{m_{ij}^k} - c_{ij}\right|}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} - \frac{\underline{m_{ij}^k} - c_{ij}}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}}\right] \tag{5}$$

$$\overline{n_{ij}^k} = \frac{1}{2}\left[\frac{\left|\overline{m_{ij}^k} - c_{ij}\right|}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} + \frac{\overline{m_{ij}^k} - c_{ij}}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}}\right] \tag{6}$$

$$\overline{p_{ij}^k} = \frac{1}{2}\left[\frac{\left|\overline{m_{ij}^k} - c_{ij}\right|}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} - \frac{\overline{m_{ij}^k} - c_{ij}}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}}\right] \tag{7}$$

$$k = 1, \cdots, m$$

where $\underline{n_{ij}^k}$ and $\underline{p_{ij}^k}$ measure the relative under-achievement and the relative over-achievement with respect to the low target $\underline{m_{ij}^k}$, respectively. Variables $\overline{n_{ij}^k}$ and $\overline{p_{ij}^k}$ play the same role for the high target $\overline{m_{ij}^k}$. They quantify in terms of the extremes of the intervals, "how relatively far" the solution c_{ij} is from the interval target for the expert k.

Because we consider interval target for each (i, j), the unwanted deviation variables are $\underline{n_{ij}^k}$ and $\overline{p_{ij}^k}$ and they have to be minimized. Therefore, for each pair (i, j) $(i \neq j)$, we look for the c_{ij} data that minimizes the objective function:

$$\left[\sum_{k=1}^{m}\left(\underline{n_{ij}^k} + \overline{p_{ij}^k}\right)^p\right] \quad if\ 1 \leq p < \infty \tag{8}$$

subject to

$$\frac{\overline{\underline{m_{ij}^k}} - c_{ij}}{\log \overline{\underline{m_{ij}^k}} - \log \underline{m_{ij}^k}} - \underline{n_{ij}^k} + \underline{p_{ij}^k} = 0 \qquad k = 1, ..., m \tag{9}$$

$$\frac{\overline{m_{ij}^k} - c_{ij}}{\log \overline{m_{ij}^k} - \log \underline{m_{ij}^k}} - \overline{n_{ij}^k} + \overline{p_{ij}^k} = 0 \qquad k = 1, ..., m \tag{10}$$

$$\underline{n_{ij}^k}, \overline{n_{ij}^k}, \underline{p_{ij}^k}, \overline{p_{ij}^k} \geq 0 \qquad k = 1, ..., m$$
$$c_{ij} > 0$$

Expressions (9) and (10), connecting the variable c_{ij} with the new variables $\underline{n_{ij}^k}$, $\underline{p_{ij}^k}$, $\overline{n_{ij}^k}$ and $\overline{p_{ij}^k}$, have been obtained by substracting (4) from (5), and substracting (6) from (7), respectively.

For $p = \infty$, for each pair (i, j) $(i \neq j)$, we get a mathematical programming problem $min\ \ D_{ij}$ over the nonnegative numbers subject to the above goals and constraints plus $\underline{n_{ij}^k} + \overline{p_{ij}^k} \leq D_{ij}, k = 1, ..., m$. D_{ij} is an extra nonnegative variable that quantifies the maximum relative deviation for the (i, j)-entry.

For the most common values of p, $p = 1$ and $p = \infty$, the above formulations are reduced to linear programming problems that can be solved using the simplex method. The case $p = 2$ is a quadratic programming problem for which several numerical tools are available.

Example 1. Let us present a numerical example ([17]) to illustrate how the proposed methodology works. A group of four experts assess their preferences about four alternatives, through the pc interval matrices M^1, M^2, M^3 and M^4, on the Saaty's scale ([14]) as follows:

$$M^1 = \begin{pmatrix} [1,1] & [5,9] & [\frac{1}{5}, \frac{1}{3}] & [3,5] \\ [\frac{1}{9}, \frac{1}{5}] & [1,1] & [\frac{1}{9}, \frac{1}{5}] & [\frac{1}{7}, \frac{1}{5}] \\ [3,5] & [5,9] & [1,1] & [3,7] \\ [\frac{1}{5}, \frac{1}{3}] & [5,7] & [\frac{1}{7}, \frac{1}{3}] & [1,1] \end{pmatrix} \quad M^2 = \begin{pmatrix} [1,1] & [1,3] & [3,5] & [3,5] \\ [\frac{1}{3}, 1] & [1,1] & [\frac{1}{7}, \frac{1}{3}] & [\frac{1}{5}, \frac{1}{3}] \\ [\frac{1}{5}, \frac{1}{3}] & [3,7] & [1,1] & [1,5] \\ [\frac{1}{5}, \frac{1}{3}] & [3,5] & [\frac{1}{5}, 1] & [1,1] \end{pmatrix}$$

$$M^3 = \begin{pmatrix} [1,1] & [3,5] & [\frac{1}{5}, \frac{1}{3}] & [3,5] \\ [\frac{1}{5}, \frac{1}{3}] & [1,1] & [\frac{1}{5}, \frac{1}{3}] & [\frac{1}{7}, \frac{1}{3}] \\ [3,5] & [3,5] & [1,1] & [5,7] \\ [\frac{1}{5}, \frac{1}{3}] & [3,7] & [\frac{1}{7}, \frac{1}{5}] & [1,1] \end{pmatrix} \quad M^4 = \begin{pmatrix} [1,1] & [3,5] & [1,3] & [5,7] \\ [\frac{1}{5}, \frac{1}{3}] & [1,1] & [\frac{1}{5}, \frac{1}{3}] & [\frac{1}{5}, \frac{1}{3}] \\ [\frac{1}{3}, 1] & [3,5] & [1,1] & [3,5] \\ [\frac{1}{7}, \frac{1}{5}] & [3,5] & [\frac{1}{5}, \frac{1}{3}] & [1,1] \end{pmatrix} \tag{11}$$

We find that matrices given by the experts provide information discrepant and not compatible.

Table 1. Consensus matrix C for $p = 1, 2, \infty$

Matrix C for $p = 1$	Matrix C for $p = 2$	Matrix C for $p = \infty$
$\begin{pmatrix} 1.000 & 5.000 & 0.332 & 5.000 \\ 0.201 & 1.000 & 0.200 & 0.200 \\ 2.999 & 5.000 & 1.000 & 5.000 \\ 0.200 & 5.000 & 0.200 & 1.000 \end{pmatrix}$	$\begin{pmatrix} 1.000 & 4.555 & 1.221 & 5.000 \\ 0.231 & 1.000 & 0.200 & 0.200 \\ 2.036 & 5.000 & 1.000 & 5.000 \\ 0.200 & 5.000 & 0.200 & 1.000 \end{pmatrix}$	$\begin{pmatrix} 1.000 & 4.303 & 1.666 & 5.000 \\ 0.247 & 1.000 & 0.200 & 0.200 \\ 1.666 & 5.000 & 1.000 & 5.000 \\ 0.200 & 5.000 & 0.200 & 1.000 \end{pmatrix}$

The proposed method is applied in order to find the consensus matrix C (first phase) using l_p-metrics, for $p = 1$, $p = 2$ and $p = \infty$ and the results are listed in Table 1.

3 Generating the Group Preference Weights

Once the l_p-group preference matrix C has been computed, the task is to obtain the crisp preference weights $w_1, ..., w_n$, of the alternatives from the matrix C. We assume that preference weights are positive and normalized, i.e. $\sum_{i=1}^{n} w_i = 1$.

Several procedures are available in the literature concerning this problem. The eigenvector method [14] is the standard method employed in the AHP context for reciprocal matrices. On the other hand, distance-based methods are provided by [15] and [16] among others. We adopt the distance-based approach followed in [16]. The idea is to look for $w = (w_1, ..., w_n)^t$ taking into account the consistency properties of the matrix C in a l_q-distance framework. The priority vector is obtained by solving the following optimization problem:

$$\min_{w \in F} \left[\sum_{i=1}^{n} \sum_{j=1}^{n} |c_{ij} w_j - w_i|^q \right]^{1/q} \qquad if \ 1 \leq q < \infty \tag{12}$$

$$\min_{w \in F} \left[\max_{i,j=1,..,n} (|c_{ij} w_j - w_i|) \right] \qquad if \ q = \infty$$

over the feasible set $F = \{w = (w_1, ..., w_n)^t / w_i > 0, \sum_{i=1}^{n} w_i = 1\}$.

We apply the second phase of the methodology to compute the preference weights associated to the matrices of the example given in section 2. For simplicity's sake, we assume the matrix C has been obtained with $p = 2$. The obtained priority vectors and the associated rankings for the most usual values of q, $q = 1, 2, \infty$ are listed in Table 2.

We notice that there is a tie between options x_2 and x_4 for metric $q = 1$. This tie is solved in the results obtained with $q = 2$ and $q = \infty$ yielding dominance for option x_4 over x_2. We remark that as q increases, the effect of greater deviations is emphasized.

Table 2. Preference weights and their associated rankings for $q = 1, 2, \infty$

	Metric		
	$= 1$	$= 2$	$= \infty$
	$(0.394, 0.087, 0.432, 0.087)^t$	$(0.357, 0.067, 0.483, 0.093)^t$	$(0.345, 0.069, 0.471, 0.115)^t$
Ranking	$_3 \succ \ _1 \succ \ _4 = \ _2$	$_3 \succ \ _1 \succ \ _4 \succ \ _2$	$_3 \succ \ _1 \succ \ _4 \succ \ _2$

We developed a GP matrix generator using MS Visual FoxPro® and problems were optimized using ILOG CPLEX® (Java classes).

4 Conclusions

Methods for learning and predicting preferences in an automatic way is a topic in disciplines such as machine learning, recommendation systems and information retrieval systems. A problem in this field is the scoring and ranking of decision alternatives from imprecise preference information from different sources. We focus on the problem of learning the overall preference weights of a set of alternatives in a multiple interval pc matrices scenario.

The proposed methodology is articulated into two phases. First, it is provided a l_p-distance model attending to data precision, that synthesizes expert's interval pc matrices into a crisp group matrix. This phase may prove to be useful in a group decision problem where difficulties in articulating consensus information from conflicting interests and different viewpoints are most common. In this context the parameter p has a consensus meaning, it places more or less emphasis on the relative contribution of individual deviations. On the other hand, the effect of the precision degree, attached to deviations in the objective function considered in the paper, is to place more or less emphasis on the relative contribution of data deviations according to the precise knowledge of the data.

In the second phase, we deal with the problem of outputting crisp weights of the alternatives from the group information. Most of the methods dealing with interval data lead to interval weights. When the interval weights overlap, there is no unique ranking of alternatives. In this case, additional ranking procedures are required in order to compare the final alternative scores.

Another key characteristic of our approach is the ability of numerical methods for computing the proposed solution for the most usual metrics.

Acknowledgements. The authors would like to thank to the anonymous reviewers for their valuable comments and suggestions. This research has been partly supported by Project MTM2007-67232, "Ministerio de Educación y Ciencia" of Spain.

References

1. Doyle, J.: Prospects for preferences. Computational Intelligence 20(2), 111–136 (2004)
2. Fürnkranz, J., Hüllermeier, E.: Pairwise preference learning and ranking. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 145–156. Springer, Heidelberg (2003)
3. Arbel, A., Vargas, L.: Preference simulation and preference programming: Robustness issues in priority derivation. European Journal of Operational Research 69(2), 200–209 (1993)
4. Salo, A.A., Hämäläinen, R.P.: Preference programming through approximate ratio comparisons. European Journal of Operational Research 82, 458–475 (1995)
5. Wang, Y.-M., Yang, J.-B., Xu, D.-L.: Interval weight generation approaches based on consistency test and interval comparison matrices. Applied Mathematics and Computation 167(1), 252–273 (2005)
6. Islam, R., Biswal, M.P., Alam, S.S.: Preference programming and inconsistent interval judgments. European Journal of Operational Research 97(1), 53–62 (1997)
7. Wang, Y.-M., Yang, J.-B., Xu, D.-L.: A two-stage logarithmic goal programming method for generating weights from interval comparison matrices. Fuzzy Sets and Systems 152, 475–498 (2005)
8. Boender, C.G.E., de Graan, J.G., Lootsma, F.A.: Multicriteria decision analysis with fuzzy pairwise comparisons. Fuzzy Sets and Systems 29(2), 133–143 (1989)
9. Yu, C.-S.: A GP-AHP method for solving group decision-making fuzzy AHP problems. Computers and Operations Research 29, 1969–2001 (2002)
10. Mikhailov, L.: Group prioritization in the AHP by fuzzy preference programming method. Computers and Operations Research 31(2), 293–301 (2004)
11. Bryson, N.K.M., Joseph, A.: Generating consensus priority interval vectors for group decision-making in the AHP. Journal of Multi-Criteria Decision Analysis 9(4), 127–137 (2000)
12. González-Pachón, J., Romero, C.: Aggregation of partial ordinal rankings: an interval goal programming approach. Computers and Operations Research 28(8), 827–834 (2001)
13. Ignizio, J.: Linear programming in single -& multiple- objective systems. Prentice-Hall, Englewood Cliffs (1982)
14. Saaty, T.: The analytic hierarchy process. McGraw-Hill, New York (1980)
15. Jones, D.F., Mardle, S.J.: A distance-metric methodology for the derivation of weights from a pairwise comparison matrix. Journal of the Operational Research Society 55, 869–875 (2004)
16. Dopazo, E., González-Pachón, J.: Consistency-driven approximation of a pairwise comparison matrix. Kybernetyca 39(5), 561–568 (2003)
17. Dopazo, E., Ruiz-Tagle, M.: A GP formulation for aggregating preferences with interval assessments. In: 7th International Conference on Multiobjective Programming and Goal Programming (submitted, 2006)

Psychometric Functions Within the Framework of Binary Signal Detection Theory: Coding the Face Identity

Petro Gopych[1] and Anna Kolot[2]

[1] Universal Power Systems USA-Ukraine LLC, 3 Kotsarskaya st., Kharkiv 61012, Ukraine
[2] Prof. L.L. Girshman Municipal Clinic no. 14, 5 Oles Gonchar st., Kharkiv 61023, Ukraine
pmg@kharkov.com

Abstract. One of standard methods in vision research is measuring the psychometric functions (PFs) that are further analyzed implying the validity of traditional signal detection theory (SDT). This research paradigm contains essential inherent contradiction: in contrast to most empirical PFs the ones predicted by the SDT do not satisfy the Neyman-Pearson objective. The problem may successfully be overcome within the framework of recent binary signal detection theory (BSDT) providing PFs for which the objective required is always achieved. Here, the original BSDT theory for vision is for the first time applied to quantitative description of specific empirical PFs measured in experiments where the coding of facial identity has been studied. By fitting the data, some parameters of BSDT face recognition algorithm were extracted and it was demonstrated that the BSDT supports popular prototype face identification model. Results can be used for developing new high-performance computational methods for face recognition.

Keywords: neural networks, generalization through memory, Neyman-Pearson objective, face recognition, prototype face identification model.

1 Introduction

Psychometric functions (PFs, hit rates vs. stimulus magnitudes [1,2]) are widely used in vision research, e.g., [3-8]. Implying (often implicitly) the validity of traditional signal detection theory (SDT) [1,2], they are further quantitatively described by two parameters: empirical stimulus threshold and the slope of PF curve at the point of subjective equality (the stimulus for which responses "yes" occur on 50% of trials). Such a consideration, in spite of its almost ubiquitous use, is essentially inappropriate to most psychophysics (including vision) experiments because subjects are usually instructed, explicitly or implicitly, when evaluating stimuli of different magnitudes to keep constant a given false-alarm rate — the goal (so-called Neyman-Pearson objective [2]) that for SDT PFs can never be achieved even in principle. This often-ignored fact hinders the complete theoretical description of observed PFs and the discovering of underlying processes that generate their specific shapes.

This SDT fundamental drawback can be overcome using the recent binary signal detection theory (BSDT [9-14]) because its PFs satisfy the Neyman-Pearson objective

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 248–257, 2007.

and the BSDT provides its own parameter (false-alarm probability, *F*), specifying the measured PFs and reflecting the fact that the objective required is always achieved.

In this paper, as a development of (or an alternative to) the SDT, we introduce a BSDT description of empirical PFs and compare the both approaches' similarities and distinctions, advantages and disadvantages. The BSDT theory for vision [12] allowing generalization from the small number of examples or even from a single example is applied to the BSDT explanation of PFs measured in face identity experiments. It has been demonstrated that BSDT qualitative and quantitative predictions are consistent with empirical data [6,7] and a popular prototype face identification model [3-8].

2 Disadvantages of SDT Psychometric Functions

The SDT defines two quite separate spaces: the stimulus space where all an animal's input stimuli are being processed and a hypothetical decision space where stimulus magnitudes appear as locations (usually distributed as a Gaussian) on a decision parameter axis. On the one hand, the distinctness of these spaces is the SDT's advantage, because it allows to remain out of considering the problem of discovering specific mechanisms of input stimulus transduction into variables in decision space and to focus on solving, on the basis of mathematics and probability theory, decision-making problems in the space deliberately designed for this purpose. On the other hand, that is the SDT's disadvantage simultaneously because relations of input stimuli to their decision space representations (which are of essential interest) can be never discovered in principle as the nature of these representations is unknown and unspecified *by definition*. In spite of that, the SDT explicitly implies that stimulus values are monotonically (or even linearly) related to the values of their decision space representations [2]. As PFs are functions of stimulus magnitude, for their shape descriptions, that is a mandatory requirement (see Fig. 1C and D).

In order to explain the results of psychophysics experiments (hit rates measured, *H*, and false alarms, *F*), two basic parameters are introduced in SDT decision space [1,2]: sensitivity, *d'* (the ability to discriminate), and bias, *c* (a tendency to use the response irrespective to the stimulus presented). SDT performance functions are shown in Fig. 1: hit rates vs. sensitivity or basic decoding performance, BDP, curves (A); hit rates vs. false-alarm rate or receiver operating characteristic, ROC, curves (B); hit rates vs. stimulus magnitude or psychometric function, PF, curves (C); sensitivity vs. stimulus magnitude or a model for input stimulus signal transduction (D).

BDPs and ROCs are *universal* functions, not depending on the stimulus model. PFs are the BDPs rescaled under the assumption of validity of a stimulus model that produces particular functions relating the stimulus and its representation (the simplest examples are in Fig. 1D). As for each BDP the Neyman-Pearson objective is not achieved (Fig. 1A), the same holds for any PF based on it. Hence, SDT PFs (as in Fig. 1C) cannot be used for explaining most of empirical PFs even if their shapes would coincide completely. Simultaneously, SDT statistics for the parametrization of empirical PFs (the threshold, PSE, jnd or the slope of PF curve, Fig. 1C) retain their initial sense because they were defined only assuming the specific ogive shape of SDT (and empirical) PFs. The latter would perhaps excuse SDT fundamental inability of describing empirical PFs and explain why this fact is so often ignored.

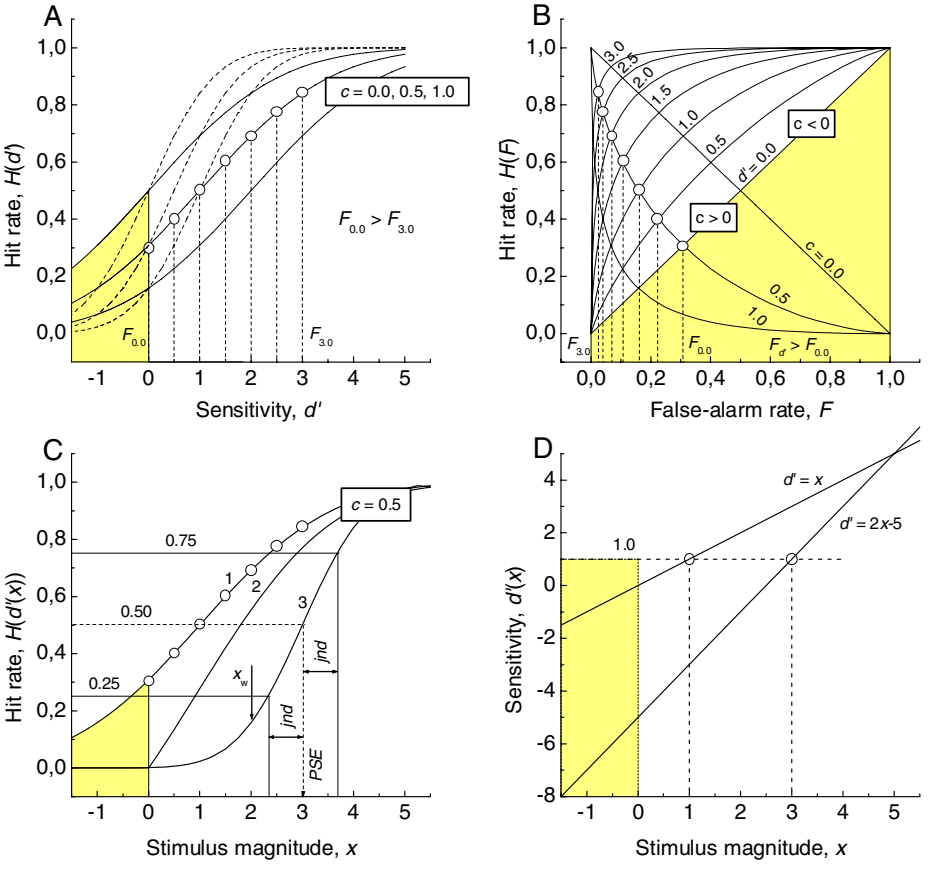


Fig. 1. SDT performance functions. **A**, BDPs (isobias curves) for criteria $c = 0.0$, 0.5, and 1.0; dashed and solid curves, cumulative distribution functions for Gaussians with arguments $z = d'$ and $z = 2d'$, respectively (for drawing the left-most dashed curve, Table 5A.2 in [2] can be used); circles on the curve $c = 0.5$, $z = 2d'$ are the same as in **B**; for each circle, its specific false alarm, $F_{d'}$, may be readout from horizontal axis in **B** ($F_{0.0} > F_{0.5} > F_{1.0} > F_{1.5} > F_{2.0} > F_{2.5} > F_{3.0}$; $F_{d'} \to 0$ as $d' \to +\infty$ and $F_{d'} \to 1$ as $d' \to -\infty$; hence, here the Neyman-Pearson objective is not achieved). **B**, ROCs (isosensitivity curves) for sensitivities $d' = 0.0$, 0.5, 1.0, 1.5, 2.0, 2.5, and 3.0 (Eq. 1.8 in [2]) and isobias curves for $c = 0.0$, 0.5, and 1.0 (Eq. 2.7 in [2]); circles (the same as in **A**) indicate cross-points of all the ROCs shown with the curve $c = 0.5$ (dashed lines point to their false alarms, $F_{d'}$). **C**, PFs (isobias curves) for $c = 0.5$ and given linear stimulus transduction models shown in **D**; curves 1 and 2, $d'(x) = x$ (curve 2 is curve 1 corrected for guessing, Eq. 11.6 in [2]); curve 3, $d'(x) = 2x - 5$; jnd, just noticeable difference (in a similar way for curve 1, jnd cannot be found because negative x will be involved); PSE, the point of subjective equality or an empirical threshold; x_w, the weakest detectable stimulus for curve 3 (another though rarely used definition of this threshold); given $d'(x)$, in a broad range of c (if $c > c_0$ where c_0 is large enough), PFs produce the same jnd and different PSEs; different linear $d'(x)$ generate different jnd and c_0. **D**, Examples of physical stimulus transduction models (cf. Fig. 5.3a in [2]), $d'(x) = x$ (cf. curves 1 and 2 in **C**) and $d'(x) = 2x - 5$ (cf. curve 3 in **C**); circles and dashed lines indicate $x = 1$ and $x = 3$ corresponding to $d' = 1$ (in **C** on curves 1 and 3 they lead to $H(x) = 0.5$). In **A**, **B**, **C**, and **D**, nonphysical areas ($d' < 0$ or $x < 0$) are shaded.

3 Advantages of BSDT Psychometric Functions

The BSDT [9-14] operates with N-dimensional vectors x with their components $x^i = \pm 1$, a reference vector $x = x_0$ representing the information stored or that should be stored in a neural network (NN), binary noise $x = x_r$ (the signs of its ±1 components are randomly chosen with uniform probability, ½), and vectors $x(d)$ with components

$$x_i(d) = \begin{cases} x_0^i, & if \quad u_i = 0, \\ x_r^i, & if \quad u_i = 1 \end{cases} \quad d = \sum u_i / N, \quad i = 1,..., N \tag{1}$$

where u_i is 0 or 1. If m is the number of marks $u_i = 1$ then $d = m/N$, $0 \le d \le 1$; d is a fraction of noise components in $x(d)$, $q = 1 - d$ is a fraction of intact components of x_0 in $x(d)$ or an intensity of the cue (cue index), $0 \le q \le 1$. If $d = m/N$, the number of different $x(d)$ is $2^m C^N_m$, $C^N_m = N!/(N-m)!/m!$; if $0 \le d \le 1$, this number is $\sum 2^m C^N_m = 3^N$ ($m = 0, 1, \ldots, N$). As the set of $x(d)$ is complete, always $x = x(d)$.

BSDT decoding algorithm exists in NN, convolutional, and Hamming forms that are equivalent and the best in the sense of pattern recognition quality [13,14]; all its predictions are essentially discrete. For intact perfectly learned NNs, the hit rate, H (or correct decoding probability of vectors x, P), can be calculated analytically [9,14]:

$$P(N,m,\Theta) = \sum_{k=0}^{K} C_k^m / 2^m, \quad K_0 = \begin{cases} (N-\Theta-1)/2, & if \quad N \quad is \quad odd \\ (N-\Theta)/2-1, & if \quad N \quad is \quad even \end{cases} \tag{2}$$

where Θ is an even integer θ, $-N \le \Theta < N$ (θ is the neuron's triggering threshold, $-\infty < \theta < +\infty$); if $K < K_0$ then $K = m$ else $K = K_0$ (k, the Hamming distance from $x = x(d)$ to x_0; K, its threshold value; K_0, the K for a given Θ). If $\Theta < -N$ then $P(N,m,\Theta) = 1$, if $\Theta \ge N$ then $P(N,m,\Theta) = 0$. For any $\theta \in \Delta\theta_j$, $P(N,m,\theta) = P(N,m,\Theta_j)$ where $\Theta_j \in \Delta\theta_j$ (here, $j = 0, 1, 2, \ldots, N+1$; $\Theta_j = 2j - N - 1$). If $0 < j < N + 1$ then $\Delta\theta_j = [\Theta_j - 1, \Theta_j + 1)$ and $\Delta\theta_j = [\Theta_j, \Theta_j + 2)$ for odd and even N, respectively; if $j = 0$ and $N + 1$ then $\Delta\theta_0 = (-\infty, -N)$, $\Delta\theta_{N+1} = [N, +\infty)$ and $P(N,m,\Theta_0) = 1$, $P(N,m,\Theta_{N+1}) = 0$ [9,10].

For a given N, $P(N,m,\Theta_j) = P(N,q,j) = P(q,F) = H(q,F)$ where j is the confidence level of decisions [13] and $F = F_j$ (F, false-alarm probability). If q (cue index) is fixed then $P(q,F) = P_q(F) = H(F)$, this function is called receiver operating characteristic (ROC) curve; if F is fixed then $P(q,F) = P_F(q) = H(q)$, this function is called basic decoding performance (BDP) curve. ROCs and BDPs are fundamentally discrete-valued although as $N \to \infty$ they tend to become continuous. ROCs at all q and BDPs at all F provide the same information [the lattice of $(N + 1) \times (N + 2)$ values of $P(q,F) = H(q,F)$]. Examples of BDPs and ROCs are in Fig. 2A and B, respectively.

As BSDT BDPs [or $P(q,F) = H(q)$] are defined at a given F, for them the Neyman-Pearson objective is achieved *by definition*. For this reason, theoretical BDPs (Fig. 2A) may naturally be used for explaining empirical PFs (Fig. 2C) and in this case, in contrast to the SDT, any contradiction between theoretical and empirical assumptions does not appear. The fruitfulness of this approach was earlier demonstrated by the comparison of theoretical BDPs found for a BSDT NN local feature discrimination algorithm (peak search code *PsNet* [15]) to appropriate empirical data in humans [16] (in fact, they coincide completely, ref. 15 and its Fig. 6).

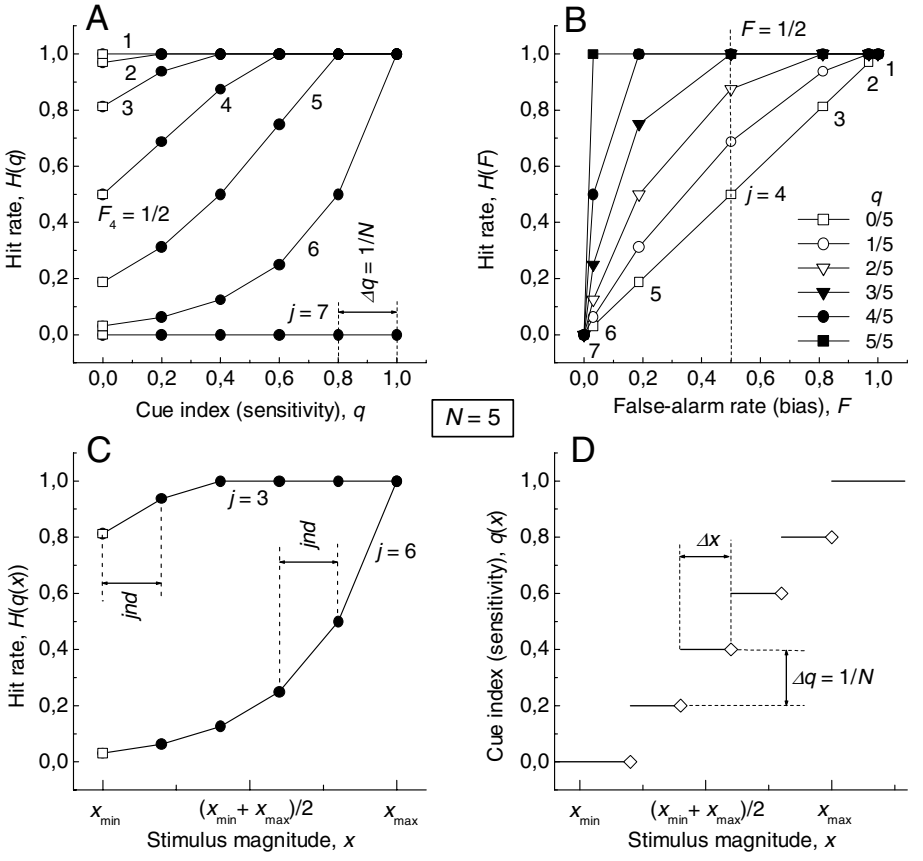


Fig. 2. BSDT performance functions. **A**, Complete set of $N + 2$ BDPs given N (signs connected by strait-line segments); j, the confidence level of decisions (the BDP's series number) [13]; F_j (open square), false alarm or 'bias' specific to the jth BDP (hence, in this case the Neyman-Pearson objective is always achieved); Δq, minimal cue-index interval (a BSDT dimensionless counterpart to the jnd). **B**, Complete set of $N + 1$ ROCs given N (signs connected by strait-line segments); each ROC corresponds to its specific cue index, q ($q = 1 - d$ is the BSDT's sensitivity measure [10], ROCs and their cue indices are related in the insertion); dashed line indicates hit rates for $F = 1/2$ (the same as along curve 4 in **A**). **C**, Two ($j = 3$ and 6) of $N + 2$ possible PFs for the stimulus transduction model shown in **D** (designations as in **A**, other PFs are not shown in order to not overload the picture); $jnd = (x_{\max} - x_{\min})/N$, just noticeable difference (the PSE is not here required; cf. Fig. 1C). **D**, The simplest example of a possible model, $q(x)$, of the stimulus' physical signal transduction into the activity of decision-making neurons — a set of horizontal line segments defined by the rules: if $x \in (-\infty, x_1)$ then $q(x) = 0$, if $x \in [x_k, x_{k+1})$ then $q(x) = k/N$ ($k = 1, \ldots, N - 1$), if $x \in [x_N, +\infty)$ then $q(x) = 1$; $x_k = k\Delta x$ ($k = 1, \ldots, N$), $\Delta x = (x_{\max} - x_{\min})/N$; $q(x)$ is a piecewise constant single-valued function, in points of discontinuity open diamonds designate where $q(x)$ is not defined (cf. [11,14]); Δq, minimal cue-index interval (the same as in **A**). Hit rates were analytically calculated (Eq. 2) for a perfectly learned intact NN of the size $N = 5$ (it may be an apex NN for the learned NN hierarchy related to a neural subspace [12], in particular, to a face neural subspace discussed in Section 4).

Another BSDT advantage is the existence of a tool (a *neural space* [10,12]) for the relating of particular stimulus signals to the activity of particular decision-making neurons or, in other words, for discovering in principle the function $q(x)$ (Fig. 2D) on the basis of neuroscience information. BSDT neural space, which is common for input stimuli and decisions concerning them, is a hierarchy of universal NN units of known structure each of which is learned to recognize optimally its own reference pattern x_0 [12,14] though, as final decisions concerning particular stimulus processing are made by the NN (NN memory unit) sitting at the apex of particular hierarchy, the circuits consuming processing results see the apex's NN output only (cf. Fig. 2) and have no direct information on the hierarchy itself. For this reason, for the description of a stimulus, it is often enough to use its apex NN and x_0 stored in it (in such cases, for external observers, the hierarchy's internal structure is not essential in general). Because each type of inputs is being processed by its specific hierarchy of learned NNs, we refer to this neuron space fraction as a *neural subspace* (or a *semi-representational* trace of memory [12] dealing with particular type of stimuli and exploring a kind of predictive coding [17])—it should be purposefully designed in machines or anatomically predefined in living organisms. Hence, BSDT neural space (the whole semi-representational memory) consists of a huge amount of overlapping neural subspaces (semi-representational memory traces) constituting a milieu where (brain) input codes (stimuli) are being processed and decisions are being made.

4 BSDT Face Neural Subspace, FNS

Generalization problem is traditionally considered in the context of the classic learning theory as a problem of the best approximate interpolation of some data points ('examples') by a rather smooth continuous function—generalization from examples [18]. The more the number of examples the more productive this approach is while if there is the only example then it does not work at all. It is implicitly assumed that the transition from one empirical point (or image) to any other one is being performed continuously, through an *infinite* number of other intermediate points. As memories of infinite capacity are impossible, such intermediate patterns are calculated by interpolating among examples. The BSDT offers another approach—generalization in discrete finite dimensional spaces (Fig. 3) or through NN memories (such as apex NNs mentioned above) each of which stores a single binary pattern x_0 only [12].

Because of Eq. 1 each $x(d)$ may be interpreted as specific x_0 damaged by noise to the damage degree $d = m/N$. For this reason, BSDT BDPs and ROCs (as in Fig. 2A and B) describe also *generalization ability* of an NN memory unit sitting at the apex of a NN hierarchy concerning particular neural subspace—the probability (Eq. 2) of interpreting $x(d)$ as x_0 or retrieving the x_0 if the retrieving process is initiated by $x = x(d)$. This very special role of $x_0 = x(0)$ provides the BSDT's possibility to generalize even from a single example, x_0 (because any $x(d)$ may be considered as a noised version of x_0, see also Eq. 7 in [12]). Generalization through memory also explores the idea of optimal recognition of a pattern of any complexity if it is presented in a semi-representational way, as its specific hierarchy of learned NN units (i.e., as a BSDT neural subspace [10,12]). The rules required for designing the hierarchy are so far explicitly not known but, as we can see, in many cases this fact is not essential.

Without any loss of generality, here we say of one specific sort of patterns only, human faces (Fig. 3). If so, reference face (RF) x_0 of a BSDT face neural subspace (FNS) is a direct counterpart to a stored in memory norm, prototype, mean/average, or abstract face against which, as it is assumed in current vision theory, the face identity is coded and which is dynamically tuned by experience [3-8]. For details see Fig. 4.

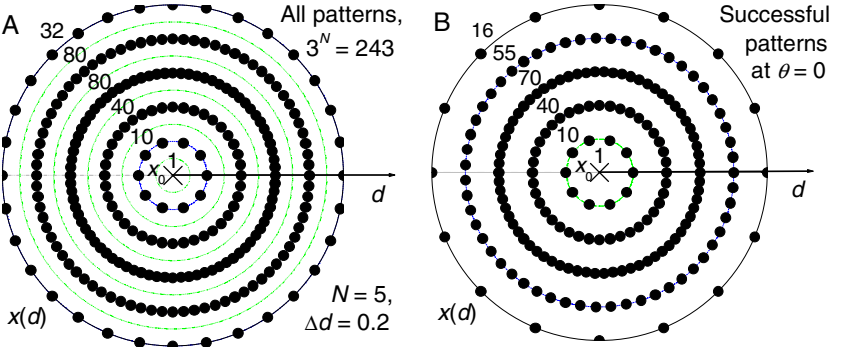


Fig. 3. BSDT N-dimensional (N = 5) face neural subspace (FNS) corresponding to reference face (RF) x_0. **A**, Complete set of all the faces $x(d)$ (the versions of x_0) constituting the FNS; cross in the center, RF x_0 [$x(d)$ at d = 0, $x_0 = x(0)$]; filled points at the distance d of the cross, faces $x(d)$ given their damage degree $d = m/N$ (m = 1, …, 5; $\Delta d = 1/N$); numbers, the amount ($2^m C^N_m$) of different $x(d)$ given d. **B**, Complete set of all successful (successfully recognizable) faces at θ = 0 (θ, neuron triggering threshold for an apex NN; other designations as in **A**); numbers, the amount of successful faces $x(d)$ given d and θ found by Eq. 2 (data are the same as in Fig. 2A, curve 4). Each face of the FNS in **A** is only designated as $x(d)$ stored in apex NN of its NN hierarchy while the hierarchy itself is not shown; at early stages of processing, different hierarchies share common neural substrates (this claim is empirically supported [4]); each $x(d) \neq x_0$ in **A** may serve as a RF for a similar FNS embedded into the initial one, in turn each $x(d) \neq x_0$ of previous FNS may be a center for a next FNS embedded into previous one, etc (such a discrete self-similar conserving the scale fractal-like hierarchy of embedded spaces may explain the existence of brain areas consisting entirely of face-selective cells [19]).

The dimensionality (N) of a BSDT FNS and the total number (3^N) of its separate faces (if embedded FNSs are not taken into account) are defined by the size of the FNS's apex NN, N (in Fig. 3 N = 5). Like the norm face of a traditional ('morphable') human face space [3-8], the RF of an FNS is also in its center (crosses in Fig. 3, circles in Fig. 4) and the more the distance from the center (RF damage degree, d) the more the face individuality is. A given d = m/N may correspond to $2^m C^N_m$ different individual faces x(d). By changing d, we change face individuality parameters and, then, the BSDT's HFRA (human face recognition algorithm) corresponding to a given FNS [12] may produce BDPs (Fig. 2A) or PFs (Fig. 2C) which are suitable for their comparison with empirical PFs [3-8] (if signal transduction model is as in Fig. 2D). The number of theoretical points along a BSDT PF is defined (fitted) by N, the size of the NN at the apex of particular FNS hierarchy. If an image x(d) contains 0% of the target face (d = 1, q = 0) then BSDT FNS, running as a HFRA, produces false-alarm recognition probability, F. As target face percentage increases (d → 0, q → 1), the recognition probability also increases approaching 1 (Fig. 2 A and C). Comparing at a

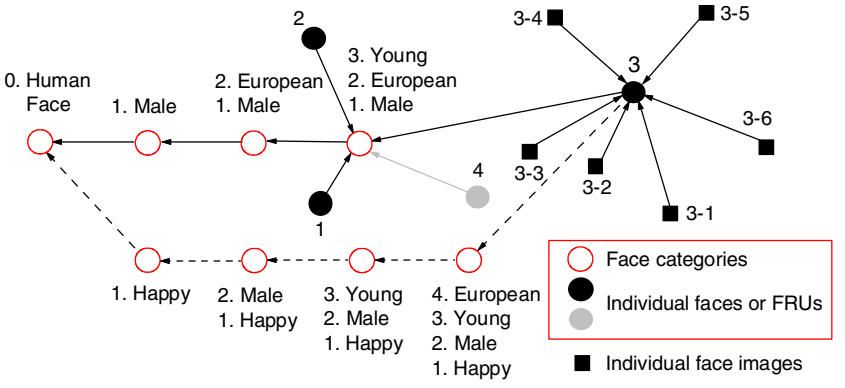


Fig. 4. Examples of possible hierarchies of embedded BSDT FNSs. Circles, RFs for the 'main' FNS (left-most circle) and for some its embedded spaces (their NN hierarchies are not shown); arrows are directed from RFs of embedded FNSs to RFs of their embodied FNSs; open and filled circles correspond to categories of faces and individual faces, respectively (the latter are counterparts to face reference units, FRUs [20]); near each open circle, its meaning is written in the number of rows that is the number of the level of embedding (e.g., 'European Male' is the RF for European-male FNS—the subspace of all encountered European-male faces, here its embedding level is 2); near each filled circle is its individual number that has nothing common with the personal name of corresponding face and is only used in this figure (circles 1 to 3, familiar faces; gray circle 4, 'unfamiliar' face having no its developed FNS). Squares, individual (e.g., view-dependent) face images (3-1 to 3-6, six such images of face 3; for faces 1 and 2, their FNSs are not displayed; RF 4 is a single individual face image ensuring generalization from this single example). Solid and dashed arrows indicate two (of many) possible embedding hierarchies for the current recognition of face 3 (in the case of success, the face is perceived as a familiar one no matter either its personal name storing in semantic memory is remembered or not; the binding of visual and semantic memory traces is here not discussed).

given F the shapes of any two empirical PFs, we may reveal which one corresponds to better (more powerful) decoding/recognition/generalization algorithm [15].

5 BSDT Explanation of Face Identity Experiments

First, we consider PFs measured along face identity trajectories 'consisting of morphs, in which the contribution of the target face is varied from 0% (no target face) to 80%' [6] (in such a way 'the face identity aftereffect for computationally opposite and non-opposite adapt-test pairs' [6] has been studied). For the parametrization instead of SDT empirical thresholds ([6] and Fig. 1C), BSDT false alarms F (recognition rate of 0% target face, Fig. 2A and C) are used, taking into account the shape of PF curves.

For opposite adapt-test pairs [6, Fig. 5 top] all PFs have a common false alarm, $F_{\text{opposite}} \approx 0.05$, similar to that for skilled subjects, $F_{\text{skill}} = 0.012 \pm 0.004$ [16]. For %Target ≥ 0, PF(Opposite Adaptation) $\geq$ PF(Baseline-Pre) and PF(Baseline-Post) $\approx$ PF(Baseline-Pre). Consequently, after the adaptation, the genuine (so far unknown but sought-for) human face recognition algorithm (HFRA) becomes uniformly more powerful and the face identity aftereffect is simply a manifestation of this

mathematical (decision theory) fact. For non-opposite adapt-test pairs [6, Fig. 5 bottom], all the PFs have probably different F similar to that typical for dilettantes, $F_{\text{dilettants}} \approx 0.3$ [16], and $F_{\text{opposite}} < F_{\text{non-opposite}}$. Very large variability of F for some individuals (e.g., MZ) says about high instability of decision criterion (or decision confidence [13]) they implicitly used. Because all F are different, corresponding PFs (quality/power functions of the HFRA) are strictly not comparable; in that case, aftereffects may be caused by the difference in F as well as the HFRA current quality. Power increasing of the HFRA, after its adaptation to an opposite face, means that opposite and test faces are on the same identity trajectory defined by a common (for this adapt-test pair) set of face identity parameters that were adjusted (tuned) in the course of the HFRA adaptation. For non-opposite adapt-test pairs, that is not the case.

For opposite pairs in three successive baseline sessions [6, Fig. 6 left], all the PFs have equal false alarms, $F \approx 0.12$, and from session to session the quality of the HFRA becomes slightly better (the small 'learning effect' [6]). For non-opposite pairs [6, Fig. 6 right], from session to session F increases from ≈ 0.12 to ≈ 0.41 (to the level of dilettantes [16]) and that is the so-called 'strong learning effect' [6].

For opposite and non-opposite trajectories in interleaved experiments [6, Fig. 7], before and after adaptation, values of F are probably the same, although $F_{\text{opposite}} < F_{\text{non-opposite}}$. After its adaptation, the HFRA becomes more powerful and that is the cause of aftereffects observed.

View-dependent face images may also be considered. In particular, poor HFRA performance (squares in Fig. 3 of ref. 7) is naturally explained assuming that, during the adaptation, the HFRA was tuned to one face while it was tested by another one (related to another FNS with another RF in its center, Figs. 3 and 4).

Of the finding that the magnitude of identity aftereffects increases as familiarity with faces increases [21] follows that familiarity enhances the HFRA's recognition power. The idea of adaptation score as a measure of familiarity effects [20] is consistent with the BSDT if PFs will be measured at a given F.

We see that empirical PFs are consistent with the notion of existence of an RF against which individual faces are coded and the BSDT HFRA [an FNS devoted to the recognition of an $x(d)$] is tuned. Adaptation to a particular face makes the HFRA more powerful, because of better matching of its parameters to individuality parameters of the adaptation face (for the lack of space, complete comparison of theoretical BSDT predictions with empirical PFs [6] is not shown). For recognizing individual faces [$x(d)$, $d > 0$, points in Fig. 3], the HFRA uses the more branched path of embedded FNSs (Figs. 3 and 4) the more the d is and, consequently, generates such neuron population responses as for empirical face tuning curves [5,8] having their minima at their RFs. Thus, the BSDT explains all the facts that traditional prototype face model explains [3-8] and even more (see legends to Figs. 3 and 4).

6 Conclusion

BSDT approach [9-14] to the description of PFs has for the first time been applied to account for PFs measured in face identity experiments. Prototype face tuning by experience and face identity aftereffects [6,7], face tuning curves [5,8], existence of brain areas consisting entirely of face-selective faces [19] agree with the BSDT which, then, might be considered as an NN substantiation of prototype face

identification model [3-8]. The results can be used for developing a new robust high-performance NN computational method for biologically plausible face analysis.

References

1. Green, D., Swets, J.: Signal Detection Theory and Psychophysics. Wiley, New York (1966)
2. Macmillan, N.A., Creelman, C.D.: Detection Theory: A User's Guide, 2nd edn. Lawrence Erlbaum Associates, Mahwah (2005)
3. Leopold, D.A., O'Toole, A.J., Vetter, T., Blanz, V.: Prototype-Referenced Shape Encoding Revealed by High-Level Aftereffects. Nature Neurosci. 4, 89–94 (2001)
4. Anderson, N.A., Wilson, H.R.: The Nature of Synthetic Face Adaptation. Vision Res. 45, 1815–1828 (2005)
5. Loffer, G., Yourganov, G., Wilkinson, F., Wilson, H.R.: fMRI Evidence for the Neural Representation of Faces. Nature Neurosci. 8, 1386–1390 (2005)
6. Rhodes, G., Jeffery, L.: Adaptive Norm-Based Coding of Facial Identity. Vision Res. 46, 2977–2987 (2006)
7. Ryu, J.-J., Chaudhuri, A.: Representations of Familiar and Unfamiliar Faces as Revealed by Viewpoint-Aftereffects. Vision Res. 46, 4059–4063 (2006)
8. Leopold, D.A., Bondar, I.V., Giese, M.A.: Norm-Based Face Encoding by Single Neurons in the Monkey Inferotemporal Cortex. Nature 442, 572–575 (2006)
9. Gopych, P.M.: ROC Curves within the Framework of Neural Network Assembly Memory Model: Some Analytic Results. Int. J. Inf. Theo. Appl. 10, 189–197 (2003)
10. Gopych, P.M.: Sensitivity and Bias within the Binary Signal Detection Theory, BSDT. Int. J. Inf. Theo. Appl. 11, 318–328 (2004)
11. Gopych, P.M.: Neural Network Computations with Negative Triggering Thresholds. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) ICANN 2005. LNCS, vol. 3696, pp. 223–228. Springer, Heidelberg (2005)
12. Gopych, P.M.: Generalization by Computation Through Memory. Int. J. Inf. Theo. Appl. 13, 145–157 (2006)
13. Gopych, P.M.: Performance of BSDT Decoding Algorithms Based on Locally Damaged Neural Networks. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 199–206. Springer, Heidelberg (2006)
14. Gopych, P.M.: Foundations of the Neural Network Assembly Memory Model. In: Shannon, S. (ed.) Leading-Edge Computer Sciences, pp. 21–84. Nova Science, New York (2006)
15. Gopych, P.M.: Identification of Peaks in Line Spectra Using the Algorithm Imitating the Neural Network Operation. Instr. Exp. Tech. 41, 341–346 (1998)
16. Gopych, P.M., Sorokin, V.I., Sotnikov, V.V.: Human Operator Performance when Identifying Peaks in a Line Spectrum. Instr. Exp. Tech. 35, 446–449 (1992)
17. Summerfield, C., Egner, T., Greene, M., Koechlin, E., Mangels, J., Hirsch, J.: Predictive Codes for Forthcoming Perception in the Frontal Cortex. Science 314, 1311–1314 (2006)
18. Poggio, T., Bizzi, E.: Generalization in Vision and Motor Control. Nature 431, 768–774 (2004)
19. Tsao, D.Y., Freiwald, W.A., Tootell, R.B.H., Livingstone, M.S.: A Cortical Region Consisting Entirely of Face-Selective Cells. Science 311, 670–674 (2006)
20. Burton, A.M., Bruce, V., Hancock, P.J.B.: From Pixels to Peoples: A Model of Familiar Face Recognition. Cog. Science 23, 1–31 (1999)
21. Jiang, F., Blanz, V., O'Toole, A.J.: The Role of Familiarity in Three-Dimensional View-Transferability of Face Identity Adaptation. Vision Res. 47, 525–531 (2007)

Load Forecasting with Support Vector Machines and Semi-parametric Method

J.A. Jordaan[1] and A. Ukil[2]

[1] Tshwane University of Technology
Staatsartillerie Road, Pretoria, 0001, South Africa
jordaan.jaco@gmail.com
[2] ABB Corporate Research
Segelhofstrasse 1K, Baden Daettwil, CH-5404, Switzerland
abhiukil@yahoo.com

Abstract. A new approach to short-term electrical load forecasting is investigated in this paper. As electrical load data are highly non-linear in nature, in the proposed approach, we first separate out the linear and the non-linear parts, and then forecast using the non-linear part only. Semi-parametric spectral estimation method is used to decompose a load data signal into a harmonic linear signal model and a non-linear trend. A support vector machine is then used to predict the non-linear trend. The final predicted signal is then found by adding the support vector machine predicted trend and the linear signal part. The performance of the proposed method seems to be more robust than using only the raw load data. This is due to the fact that the proposed method is intended to be more focused on the non-linear part rather than a diluted mixture of the linear and the non-linear parts as done usually.

1 Introduction

Short-term load forecasting (STLF) is used to estimate the electrical power demand. Accurate STLF has a significant impact on a power system's operational efficiency. Many decisions, such as spinning reserve allocation, real time generation control and security analysis, are based on STLF [1]. This also means that accurate STLF has economic and security related advantages. This allows electrical companies to commit their own production resources in order to optimise energy prices, which leads to cost savings and to increased power system security and stability [2].

In the last few years, several techniques for short- and long-term load forecasting have been discussed, such as Kalman filters, regression algorithms, artificial neural networks (ANN) [2,3] and fuzzy neural networks [1]. Another method of load forecasting is to use support vector machines (SVM). SVM is a powerful methodology for solving problems in non-linear classification, function estimation and density estimation [4]. Load forecasting is an application of function estimation (regression). In the SVM solution method one solves convex optimisation problems, typically quadratic programs with a unique solution, compared

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 258–267, 2007.

to neural network multi-layer perceptrons (MLP) where the cost function could have multiple local minima.

In [5] the authors used ANN for STLF and the training time for the ANN was quite long compared to that of SVM. For some cases the ANN performed very poorly. For the MLP, it is hard to estimate the optimal number of neurons needed for a given task [6]. This often results in over- or underfitting. This is because for MLP we choose an appropriate structure, the number of hidden layer neurons in the MLP. Keeping the confidence interval fixed in this way, we minimise the training error, i.e., we perform the empirical risk minimisation (ERM). These can be avoided using the SVM and the structural risk minimisation (SRM) principle [7]. In SRM, we keep the value of the training error fixed to zero or some acceptable level and minimise the confidence level. This way, we structure a model of the associated risk and try to minimise that. The result is the optimal structure of the SVM.

In this paper we introduce a new approach to load forecasting using SVM. Other load forecasting approaches using SVM include [7] where genetic algorithms were used in combination with SVM. The genetic algorithms were used to determine proper values for the free parameters of the SVM. In [8] the authors used regression trees to select the important input variables and to partition the input variable space for use in the SVM.

The layout of the paper is as follows: in section 2 we introduce the new proposed method of treating the load prediction problem, section 3 shows the numerical results obtained, and the paper ends with a conclusion.

2 Semi-parametric Method

When a model is fitted to the data taken from a power system, we many time have components in the data that are not directly part of the process we want to describe. If a model is fit to the data as it is, then the model parameters will be biased. We would have better estimates of the model parameters if the unwanted components (nuisance, bias, or non-linear components) are first removed. This method has been used successfully in the field of spectral estimation in power systems when we analyse the measured signals on power transmission lines [9].

The new method we propose is to separate the load data into linear and non-linear (trend) components. This method is called the Semi-Parametric method for harmonic content identification. We assume that there is an underlying linear part of the load data that could be represented with a sum of n damped exponential functions

$$y_L(k) = \sum_{i=1}^{n} A_i e^{j\theta_i} e^{(j2\pi f_i + d_i)Tk}\,, \tag{1}$$

where $y_L(k)$ is the k-th sample of the linear part of the load signal, A is the amplitude, θ is the phase angle, f is the frequency, d is the damping and T is the

sampling period. Since we work only with real signals, the complex exponential functions come in complex conjugate pairs (see eq. (16)). The equivalent Auto Regressive (AR) model of (1) is given by

$$y_L(k) = -\sum_{i=1}^{n} x_i y_L(k-i), \quad k = n+1 \ldots n+m, \tag{2}$$

with model parameters x_i, model order n, and $n+m$ number of samples in the data set. The model parameters x_i and model order n has to be estimated from the data.

We propose the following model to separate the linear and non-linear parts [9,10]:

$$y_L(k) = y(k) + \Delta y(k), \tag{3}$$

where $y(k)$ is the measured signal sample, $\Delta y(k) = E[\Delta y(k)] + \epsilon(k)$ is the residual component consisting of a non-zero time varying mean $E[\Delta y(k)]$ (nuisance or bias component) and noise $\epsilon(k)$. The mean of the residual component is represented by a Local Polynomial Approximation (LPA) model [11]. y_L is then the required linear signal that can be represented with a sum of damped exponentials (1). The LPA model is a moving window approach where a number of samples in the window are used to approximate (filter) one of the samples in the window (usually the first, last or middle sample). The LPA filtering of data was made popular by Savitsky and Golay [12,13].

By substituting eq. (3) in (2) we obtain

$$y(k) + \Delta y(k) = -\sum_{i=1}^{n} x_i [y(k-i) + \Delta y(k-i)]. \tag{4}$$

In matrix form, for $n+m$ samples, the model is

$$\mathbf{b} + \Delta\mathbf{b} = -\mathbf{A}\mathbf{x} - \Delta\mathbf{A}\mathbf{x}, \tag{5}$$

where

$$\mathbf{b} = \begin{bmatrix} y(n+1) \\ y(n+2) \\ \vdots \\ y(n+m) \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} y(n) & y(n-1) & \cdots & y(1) \\ y(n+1) & y(n) & \cdots & y(2) \\ \vdots & \vdots & \ddots & \vdots \\ y(n+m-1) & y(n+m-2) & \cdots & y(m) \end{bmatrix}, \tag{6}$$

$$\Delta\mathbf{b} = \begin{bmatrix} \Delta y(n+1) \\ \Delta y(n+2) \\ \vdots \\ \Delta y(n+m) \end{bmatrix}, \; \Delta\mathbf{A} = \begin{bmatrix} \Delta y(n) & \Delta y(n-1) & \cdots & \Delta y(1) \\ \Delta y(n+1) & \Delta y(n) & \cdots & \Delta y(2) \\ \vdots & \vdots & \ddots & \vdots \\ \Delta y(n+m-1) & \Delta y(n+m-2) & \cdots & \Delta y(m) \end{bmatrix}. \tag{7}$$

The matrix signal model (5) can be rewritten in a different form and represented as

$$\mathbf{A}\mathbf{x} + \mathbf{b} + [\Delta\mathbf{b}\ \Delta\mathbf{A}] \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix} = \mathbf{0}, \tag{8}$$

or

$$\mathbf{A}\mathbf{x} + \mathbf{b} + \mathbf{D}(\mathbf{x})\,\Delta\mathbf{y} = \mathbf{0}, \tag{9}$$

where the following transformation has been used:

$$[\Delta\mathbf{b}\ \Delta\mathbf{A}] \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix} = \mathbf{D}(\mathbf{x})\,\Delta\mathbf{y} \tag{10}$$

or

$$\left[\begin{bmatrix} \Delta y(n+1) \\ \Delta y(n+2) \\ \vdots \\ \Delta y(n+m) \end{bmatrix} \begin{bmatrix} \Delta y(n) & \Delta y(n-1) & \cdots & \Delta y(1) \\ \Delta y(n+1) & \Delta y(n) & \cdots & \Delta y(2) \\ \vdots & \vdots & \ddots & \vdots \\ \Delta y(n+m-1) & \Delta y(n+m-2) & \cdots & \Delta y(m) \end{bmatrix}\right] \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} =$$

$$\begin{bmatrix} x_n & \cdots & x_1 & 1 & 0 & \cdots & 0 \\ 0 & x_n & \cdots & x_1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & x_n & \cdots & x_1 & 1 \end{bmatrix} \begin{bmatrix} \Delta y(1) \\ \Delta y(2) \\ \vdots \\ \Delta y(n+m) \end{bmatrix}. \tag{11}$$

If the number of parameters in vector $\mathbf{x}$, (model order n) is not known in advance, the removal of the nuisance component and noise from the signal $y(k)$ is equivalent to estimating the residual $\Delta y(k)$ and the model order n while fulfilling constraints (9). To solve the semi-parametric model, the second norm of the noise, plus a penalty term which puts a limit on the size of vector $\mathbf{x}$ is minimised. The following optimisation problem should be solved:

$$\begin{aligned} \min_{\mathbf{x},\Delta\mathbf{y}} \left\{ \frac{1}{2}\|\epsilon\|_2^2 + \frac{\mu}{2}\mathbf{x}^T\mathbf{x} \right\} &= \min_{\mathbf{x},\Delta\mathbf{y}} \left\{ \frac{1}{2}(\Delta\mathbf{y} - E[\Delta\mathbf{y}])^T(\Delta\mathbf{y} - E[\Delta\mathbf{y}]) + \frac{\mu}{2}\mathbf{x}^T\mathbf{x} \right\} \\ &= \min_{\mathbf{x},\Delta\mathbf{y}} \left\{ \frac{1}{2}\Delta\mathbf{y}^T\mathbf{W}\Delta\mathbf{y} + \frac{\mu}{2}\mathbf{x}^T\mathbf{x} \right\} \end{aligned} \tag{12}$$

$$\text{subject to the equality constraints } \mathbf{A}\mathbf{x} + \mathbf{b} + \mathbf{D}(\mathbf{x})\,\Delta\mathbf{y} = \mathbf{0},$$

where

$$\mathbf{W} = (\mathbf{I} - \mathbf{S})^T(\mathbf{I} - \mathbf{S}), \tag{13}$$

$\mathbf{I}$ is the identity matrix, $\mathbf{S}$ is the LPA smoothing matrix used to estimate $E[\Delta y(k)]$ as $\mathbf{S}\Delta\mathbf{y}$, and μ is the Ridge regression factor used to control the size of vector $\mathbf{x}$ [14,15].

2.1 Estimation of the Harmonic Components

The next step is then to calculate the parameters of the harmonic components in eq. (1). We do this as follows [16,17]:

1. The coefficients x_i are those of the polynomial

$$\underline{H}(z) = 1 + \sum_{i=1}^{n} x_i \underline{z}^{-i}, \tag{14}$$

where $\underline{z}$ is a complex number

$$\underline{z} = e^{(j2\pi f + d)T}. \tag{15}$$

By determining the n roots, $\underline{z}_i$, $i = 1, 2, \ldots, n$, of eq. (14), and using eq. (15) for $\underline{z}$, we can calculate the values of the n frequencies and dampings of the harmonic components. It should be noted that we are using complex harmonic exponentials to estimate the input signal's linear component. However, the signals we measure in practice are real signals of the form

$$y_L(k) = \sum_{i=1}^{n/2} 2A_i e^{d_i Tk} cos\left(2\pi f_i Tk + \theta_i\right), \tag{16}$$

where A_i, θ_i, f_i and d_i are the same as defined for the complex harmonics in eq. (1). Therefore if we expect to have $\frac{n}{2}$ components in our real signal, there will be n complex harmonic exponentials, and thus will the AR model order be n. The complex harmonic exponentials will then always come in $\frac{n}{2}$ complex conjugate pairs.

2. To determine the n amplitudes A_i and phase angles θ_i, we substitute the linear component $y(k) + \Delta y(k)$, and the estimated frequencies and dampings into eq. (1). We obtain an overdetermined system of linear equations of $N \times n$ that can be solved using the least squares method:

$$y(k) + \Delta y(k) = \sum_{i=1}^{n} A_i e^{j\theta_i} e^{(j2\pi f_i + d_i)Tk},\ k = 1, 2, \ldots, N. \tag{17}$$

2.2 Non-linear Part

The non-linear part (plus the noise), which could represent trends or other non-linearities in the power system, is then given by

$$y_N(k) = y(k) - y_L(k), \tag{18}$$

where $y_N(k)$ is the k-th non-linear signal sample and $y(k)$ is the measured load sample. This non-linear part is then used to train a support vector machine. After the training is complete, the SVM could be used to predict the non-linear part. The linear part is calculated from the signal model (1), which is then added to the non-linear part to obtain the final predicted load values.

3 Numerical Results

For this experiment we tested many linear and non-linear SVM regression methods. For the results we show only the non-linear Radial Basis function kernel SVM. For the implementation we used MATLAB [18] and the Least Squares Support Vector Machines toolbox from [4].

Before the support vector machine is trained with the load data, some preprocessing is done on the data. First the data is normalised by dividing by the median of the data. This is because statistically the median is least affected by data variabilities. Therefore, after prediction, the signal must be de-normalised by multiplying it again with the median. Then the normalised data is separated into a linear and a non-linear part.

The test data, shown in Fig. 1, contained 29 days of load values taken from a town at one hour intervals. This gives a total number of 696 data samples. We removed the last 120 data samples from the training set. These samples would then be used as testing data. Each sample is also classified according to the hour of the day that it was taken, and according to which day. The hours of the day are from one to 24, and the days from one (Monday) to seven (Sunday).

The data fed into the support vector machine could be constructed as follows: to predict the load of the next hour, load values of the previous hours are used. We can additionally also use the day and hour information. For example, this means that as inputs to the SVM, we could have k consecutive samples, and two additional input values representing the hour and day of the predicted $k+1-th$ sample. The SVM will then predict the output of the $k+1-th$ sample. We can also call the value of k : a delay of k samples.

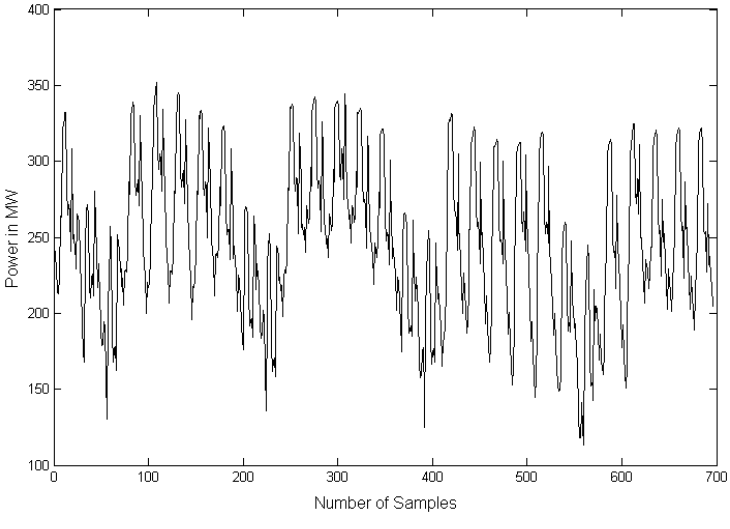


Fig. 1. Load of a Town

To evaluate the performance of the different SVMs, we define a performance index, the Mean Absolute Prediction Error (MAPE):

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|t_i - p_i|}{t_i} \times 100\,, \tag{19}$$

where t_i is the $i-th$ sample of the true (measured) value of the load, p_i is the predicted load value of the SVM, and N is the total number of predicted samples. For this experiment, the last 24 hours of the 120 removed samples in the load set was used to test the different SVMs. Different values of delay was used, from six until 96.

We also tested the prediction method without splitting the data into linear and non-linear parts, and compared it to the proposed new method. The results of the performance index for each of the methods are shown in Table 1. From the statistics in the Table it seems that the method without separating the data into different components performs slightly better than the method separating the data. In general both methods performed well, but there were occasions where the method without splitting the data had a very bad performance, eg. from delay 60 until 96. This can be seen in Fig. 2 where the bad performance is illustrated for a delay of 78. Also for a delay of 78, in Fig. 3 the method with separating the data is shown. It can be seen that this method produces better results. It was found that the method without separating the data was more sensitive to the parameters of the SVM training algorithm, than the method

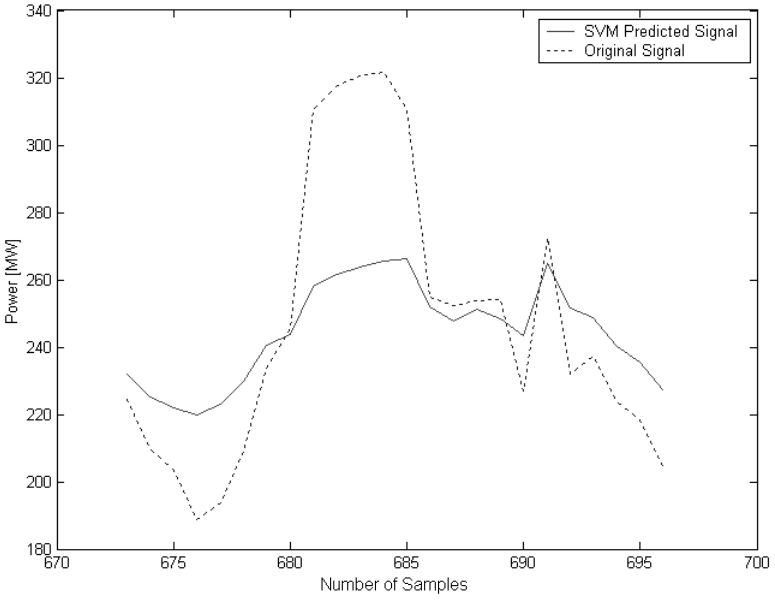


Fig. 2. Bad Performance of Method without Separating Data: Delay of 78

separating the data. The best network without splitting the data has a delay of 15 and is shown in Fig. 4.

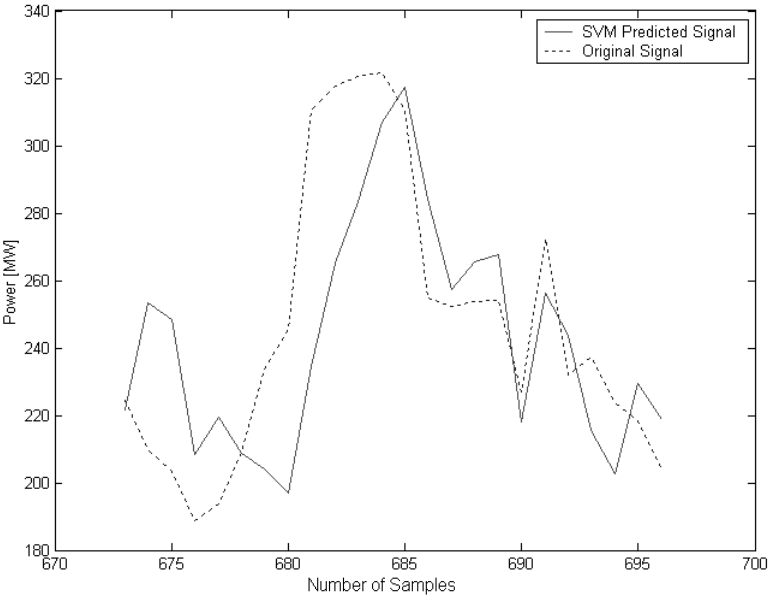


Fig. 3. Performance of Method with Separated Data: Delay of 78

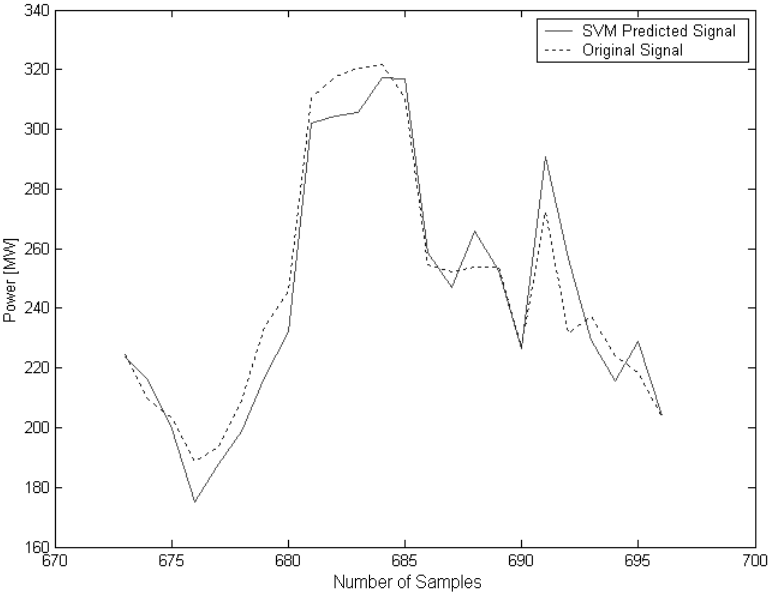


Fig. 4. Best Performance of Method with Separated Data: Delay of 15

Table 1. MAPE for the Two Methods

Delay	Separated into Linear / Non-Linear	Non-Separated
6	3.7767	2.9794
15	3.5924	3.4402
24	4.9158	4.7659
33	5.3157	5.8301
42	5.4615	5.5794
51	5.3634	4.5498
60	6.0711	5.4701
69	7.3251	6.9728
78	9.5568	8.5627
87	11.1496	10.0967
96	14.3506	11.7320
Average MAPE	6.9890	6.3617
Median MAPE	5.4615	5.5794

4 Conclusion

The Semi-Parametric method for separating the electric load into a linear and non-linear trend part was introduced. A support vector machine was then used to do load forecasting based only on the non-linear part of the load. Afterwards the linear part was added to the predicted non-linear part of the support vector machine. We compared this method to the usual method without splitting the data. On average the method without splitting the data gave slightly better results, but there were occasions where this method produced very bad results and it was also very sensitive to the SVM training parameters. The newly introduced method generally performed very well (even in the situations where the method without separating the data produced very bad results) and it was much more stable and more robust to the SVM training parameters.

This is probably due to the fact that the most important factor, the underlying non-linearities, are extracted out using the semi-parametric method and modeled using the SVM. This approach is more streamlined from the point of view of capturing the true non-linear nature of the load data by focusing only on the non-linear parts without getting diluted by taking into consideration the linear parts as is usually done.

Acknowledgement

The authors would like to thank the National Research Foundation of South Africa for their financial support.

References

1. Papadakis, S.E., Theocharis, J.B., Kiartzis, S.J., Bakirtzis, A.G.: A novel approach to short-term load forecasting using fuzzy neural networks. IEEE Transactions on Power Systems 13, 480–492 (1998)
2. Piras, A., Germond, A., Buchenel, B., Imhof, K., Jaccard, Y.: Heterogeneous artificial neural network for short term electrical load forecasting. IEEE Transactions on Power Systems 11, 397–402 (1996)
3. Bitzer, B., Rösser, F.: Intelligent Load Forecasting for Electrical Power System on Crete. In: UPEC 1997 Universities Power Engineering Conference, UMIST-University of Manchester (1997)
4. Pelckmans, K., Suykens, J., Van Gestel, T., De Brabanter, J., Lukas, L., Hamers, B., De Moor, B., Vandewalle, J.: LS-SVMlab Toolbox User's Guide, Version 1.5. Catholic University Leuven, Belgium (2003), [Online] Available from: http://www.esat.kuleuven.ac.be/sista/lssvmlab/
5. Ukil, A., Jordaan, J.: A new approach to load forecasting: Using semi-parametric method and neural networks. In: King, I., Wang, J., Chan, L., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4233, pp. 974–983. Springer, Heidelberg (2006)
6. Suykens, J.: Support Vector Machines and Kernel Based Learning. Tutorial: IJCNN, Montreal (2003), [Online], Available from: http://www.esat.kuleuven.ac.be/sista/lssvmlab/ijcnn2005_4.pdf
7. Pai, P.F., Hong, W.C.: Forecasting regional electricity load based on recurrent support vector machines with genetic algorithms. Elsevier Electric Power Systems Research 74, 417–425 (2005)
8. Yang, J., Stenzel, J.: Short-term load forecasting with increment regression tree. Elsevier Electric Power Systems Research 76, 880–888 (2006)
9. Zivanovic, R.: Analysis of Recorded Transients on 765kV Lines with Shunt Reactors. In: Power Tech2005 Conference, St. Petersburg, Russia (2005)
10. Zivanovic, R., Schegner, P., Seifert, O., Pilz, G.: Identification of the Resonant-Grounded System Parameters by Evaluating Fault Measurement Records. IEEE Transactions on Power Delivery 19, 1085–1090 (2004)
11. Jordaan, J.A., Zivanovic, R.: Time-varying Phasor Estimation in Power Systems by Using a Non-quadratic Criterium. Transactions of the South African Institute of Electrical Engineers (SAIEE) 95, 35–41 (2004), ERRATA: 94(3), 171–172 (September 2004)
12. Gorry, P.A.: General Least-Squares Smoothing and Differentiation by the Convolution (Savitzky-Golay) Method. 62, 570–573 (1990)
13. Bialkowski, S.E.: Generalized Digital Smoothing Filters Made Easy by Matrix Calculations. 61, 1308–1310 (1989)
14. Draper, N.R., Smith, H.: Applied Regression Analysis, 2nd edn. John Wiley & Sons, Chichester (1981)
15. Tibshirani, R.: Regression Shrinkage and Selection via the Lasso. Journal of the Royal Society. Series B (Methodological) 58, 267–288 (1996)
16. Casar-Corredera, J.R., Alcásar-Fernándes, J.M., Hernándes-Gómez, L.A.: On 2-D Prony Methods. IEEE CH2118-8/85/0000-0796 $1.00, 796–799 (1985)
17. Zivanovic, R., Schegner, P.: Pre-filtering Improves Prony Analysis of Disturbance Records. In: Eighth International Conference on Developments in Power System Protection, Amsterdam, The Netherlands (2004)
18. Mathworks: MATLAB Documentation - Neural Network Toolbox. Version 6.5.0.180913a Release 13 edn. Mathworks Inc., Natick, MA (2002)

Reproducing Kernel Hilbert Space Methods to Reduce Pulse Compression Sidelobes

J.A. Jordaan, M.A. van Wyk, and B.J. van Wyk

Tshwane University of Technology
Staatsartillerie Road, Pretoria, 0001, South Africa
{jordaan.jaco, mavanwyk, vanwykb}@gmail.com

Abstract. Since the development of pulse compression in the mid-1950's the concept has become an indispensable feature of modern radar systems. A matched filter is used on reception to maximize the signal to noise ratio of the received signal. The actual waveforms that are transmitted are chosen to have an autocorrelation function with a narrow peak at zero time shift and the other values, referred to as sidelobes, as low as possible at all other times. A new approach to radar pulse compression is introduced, namely the Reproducing Kernel Hilbert Space (RKHS) method. This method reduces sidelobe levels significantly. The paper compares a second degree polynomial kernel RKHS method to a least squares and L_{2P}-norm mismatched filter, and concludes with a presentation of the representative testing results.

1 Introduction

Since the development of pulse compression in the mid-1950's [1,2] the concept has become an indispensable feature of modern radar systems. Pulse compression gives radar designers the ability to obtain sufficient energy from a target for detection without degrading the range resolution of the system or resorting to the use of very high transmitter power levels. Pulse compression therefore allows for the use of lower power transmitters but with longer pulse lengths to increase the energy content of a pulse. A matched filter is used on reception to maximize the signal to noise ratio (SNR) of the received signal. The actual waveforms that are transmitted are chosen to have an autocorrelation function (ACF) with a narrow peak at zero time shift and sidelobe levels as low as possible at all other times. The sidelobes have the undesirable effect of masking smaller targets which are in close proximity to large targets, such as clutter returns. It is therefore desirable to have a main pulse lobe as narrow as possible.

This paper introduces the Reproducing Kernel Hilbert Space (RKHS) methods in radar pulse compression. It is shown that the RKHS methods reduce the sidelobe levels significantly compared to the results in [3].

In the section 2 we discuss the matched filter used in radar pulse compression and the least squares technique to solve the mismatched filter coefficients. Section 3 discusses the RKHS methods to be used in pulse compression. Results obtained in testing the algorithms are presented in section 4. The paper ends with a conclusion.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 268–276, 2007.

2 Problem Formulation

2.1 Matched Filter

In the discrete time domain the transmit pulse of a radar could be represented by a sequence of complex transmit coefficients $\{a_n\}$. Digital pulse compression is performed by the convolution of the received signal, which is assumed to be a time delayed and scaled version of the transmitted pulse, with the complex receive filter coefficients $\{z_n\}$. For the purpose of analyzing the sidelobe response of the pulse compressor, a zero time-delay and unity scaling factor can be assumed without loss of generality. In this paper a P4 code [4] (i.e. sampled linear frequency chirp) is used for the transmit coefficients.

For the transmit pulse $\{a_n\}$ the matched filter is given by $h_n = a^*_{N-n}$ where * denotes the complex conjugate and N is the number of transmit pulse samples. If the matched filter is used, the output of the pulse compressor will be the ACF of $\{a_n\}$ which is equivalent to the discrete convolution

$$b_i = \sum_i a_{i+1-k} h_k. \tag{1}$$

The convolution sequence $\{b_i\}$ for the matched filter has the maximum attainable SNR at zero time shift.

The sidelobe values of the convolution result can be minimized by introducing cost functions which map the set of sidelobes to a single real value. By minimizing the cost functions, mismatched receive filters with reduced sidelobe responses can be found. In the next section the mismatched filter solutions for the least-squares sidelobe measure, which is equivalent to the L_2-norm solution, and the generalized L_{2P}-norm are briefly discussed. In [3] the L_2- and L_{2P}-norms are presented as methods to reduce the sidelobes of the pulse compressor output. These methods will be compared to the RKHS pulse compressor.

2.2 Least Squares and L_{2P} Sidelobe Minimisation

In matrix form the output of the pulse compressor could be written as [3]

$$\mathbf{b} = \mathbf{A}_F \mathbf{z}, \tag{2}$$

where

$$\mathbf{b} = [b_1, b_2, \cdots, b_{2N-1}]^T, \tag{3}$$

$$\mathbf{z} = [z_1, z_2, \cdots, z_N]^T \tag{4}$$

and

$$\mathbf{A}_F = \begin{bmatrix} a_1 & a_2 & \cdots & a_N & 0 & \cdots & 0 \\ 0 & a_1 & a_2 & \cdots & a_N & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_1 & a_2 & \cdots & a_N \end{bmatrix}^T, \tag{5}$$

where ^{T} denotes the transpose of a vector or matrix and that $\mathbf{A}_F$ is the full convolution matrix.

The sidelobe measure function for a compressed pulse can now be formulated by defining a new matrix, $\mathbf{A}$, which is similar to $\mathbf{A}_F$, except that the row in $\mathbf{A}_F$ which produces the compression peak is removed. The sidelobe measure cost function to be minimised can therefore be written as

$$\begin{aligned} f(\mathbf{z}) &= \mathbf{b}^H\mathbf{b} \\ &= \mathbf{z}^H\mathbf{A}^H\mathbf{A}\mathbf{z} \\ &= \mathbf{z}^H\mathbf{C}\mathbf{z}, \end{aligned} \tag{6}$$

with

$$\mathbf{C} = \mathbf{A}^H\mathbf{A}, \tag{7}$$

and ^{H} denotes the complex conjugate transpose. The row in $\mathbf{A}_F$ that is removed could now be written as a constraint

$$\mathbf{a}\mathbf{z} = b_{peak}, \tag{8}$$

where $\mathbf{a} = [a_1, a_2, \cdots, a_N]$. This optimisation problem could now be solved using Lagrange multipliers.

The generalised L_{2P}-norm sidelobe cost function could now in a similar way be derived as [3]

$$\begin{aligned} f(\mathbf{z}) &= \left(\sum_{i=1}^{2N-1} \left[\|b_i\|^2 \right]^P \right)^{\frac{1}{2P}} \\ &= \left(\sum_{i=1}^{2N-1} \left[\mathbf{z}^H C_i \mathbf{z} \right]^P \right)^{\frac{1}{2P}}, \end{aligned} \tag{9}$$

where

$$C_i = \mathbf{a}_i^H \mathbf{a}_i, \tag{10}$$

and $\mathbf{a}_i$ is the ith row of $\mathbf{A}$. Using the same constraint as in (8), Lagrange multipliers could be used to solve the minimisation problem.

3 RKHS-Based Filter

The idea of a function space reproduced by a single kernel function as well as the question of whether or not there exists a kernel which will reproduce a specific function space has received attention since the beginning of the 20th century, and even before. Aronszajn [5], however, was the first to formally define the notion of a Reproducing Kernel Hilbert space during the decade 1940 to 1950.

Today the applications of the theory of reproducing kernels are widely spread in mathematical statistics and engineering applications. In the 1960's (refer to [6],[7] and [8]) Parzen applied the theory of Reproducing Kernel Hilbert spaces to

time series analysis. In the early 1970's Kailath ([9],[10],[11]) and his coworkers applied this theory to problems encountered in detection and estimation. More recently, the theory of reproducing kernel Hilbert spaces has found applications in generalised sampling theory, in wavelets and in graph matching (see [12],[13] and [14] as well as references therein).

In its simplest form a RKHS is a Hilbert space H equipped with an inner product $(\cdot,\cdot)$ and a kernel $K(\cdot,\cdot) : \mathrm{R} \times \mathrm{R} \rightarrow \mathrm{R}$ such that $K(t,\cdot) \in H$ for all $t \in \mathrm{R}$ and which has the reproducing property, i.e.

$$(F(\cdot), K(t,\cdot)) = F(t)$$

for all $t \in \mathrm{R}$. A consequence of the reproducing property is that $(K(s,\cdot), K(t,\cdot)) = K(s,t)$.

Suppose now we are given a data set of input-output training patterns $\mathcal{T} = \{t_i, f_i\}_{i=1}^{N}$ where $f_i = F(t_i) + \epsilon_i$ are noisy measurements of some unknown function $F(\cdot) : \mathrm{R} \rightarrow \mathrm{R}$. The following approximation problem is of interest: given $\mathcal{T}$ find the minimum norm approximation $\tilde{F}(\cdot)$ of $F(\cdot)$ in the RKHS H subject to the constraints $(\tilde{F}(\cdot), K(t_i,\cdot)) = f_i$. It can be shown that $\tilde{F}(\cdot)$ is of the form [15]

$$\tilde{F}(\cdot) = \sum_{i=1}^{N_c} c_i \, K(\tilde{t}_i, \cdot), \tag{11}$$

where usually $N_c \leq N$ due to the presence of noise and the kernel centres $\tilde{t}_i$ are inferred from $\mathcal{T}$ by means of some data reduction scheme [14]. The solution of this approximation problem is then obtained as

$$\mathbf{c} = \mathbf{G}^{\dagger}\mathbf{f} \tag{12}$$

where $\mathbf{c} = (a_i)$, $\mathbf{f} = (f_i)$ and $\mathbf{G} = (K(\tilde{t}_i, t_j))$. Here $\mathbf{G}^{\dagger}$ denotes the pseudo inverse of the matrix $\mathbf{G}$.

For the application discussed here we have chosen the polynomial kernel [16] namely

$$K(\mathbf{s},\mathbf{t}) = \left(1 + \mathbf{s}^T\mathbf{t}\right)^d, \tag{13}$$

where d is the degree of the polynomial. One could also use other RKHS kernels, for example the Dirichlet kernel (which is periodic) [14]

$$K(\mathbf{s},\mathbf{t}) = \frac{\sin\left(\left(n+\frac{1}{2}\right)2\pi\left(\mathbf{s}-\mathbf{t}\right)u\right)}{\sin\left(\frac{2\pi(\mathbf{s}-\mathbf{t})u}{2}\right)}, \tag{14}$$

or the *sinc* kernel

$$K(\mathbf{s},\mathbf{t}) = \frac{\sin\left(2\pi\left(\mathbf{s}-\mathbf{t}\right)u\right)}{2\pi\left(\mathbf{s}-\mathbf{t}\right)u}, \tag{15}$$

where n is the harmonic number and u is the width (or dilation) parameter of the kernel. However, for this application the polynomial kernel gave much better results, and therefore only the polynomial kernel will be considered when presenting the results.

Once the interpolator coefficients $\mathbf{c}$ are solved, we can define a template [16]

$$\tilde{\mathbf{t}} = \sum_{i=1}^{N_c} c_i \, \bar{\mathbf{t}}_i, \tag{16}$$

where

$$\bar{\mathbf{t}}_i = \left[\begin{bmatrix}1 & \mathbf{t}_i^T\end{bmatrix} \otimes \begin{bmatrix}1 & \mathbf{t}_i^T\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}1 & \mathbf{t}_i^T\end{bmatrix}\right]^T, \tag{17}$$

$\otimes$ denotes the Kronecker Tensor Product, $\mathbf{t}_i$ is an input vector, and the term $\begin{bmatrix}1 & \mathbf{t}_i^T\end{bmatrix}$ in (17) is repeated d times. For example if a second order polynomial kernel is used, then

$$\bar{\mathbf{t}}_i = \left[\begin{bmatrix}1 & \mathbf{t}_i^T\end{bmatrix} \otimes \begin{bmatrix}1 & \mathbf{t}_i^T\end{bmatrix}\right]^T. \tag{18}$$

Once the RKHS pulse compressor is trained, the template $\tilde{\mathbf{t}}$ is used to calculate the ith output of the RKHS pulse compressor as $\tilde{\mathbf{t}}^T \bar{\mathbf{t}}_i$.

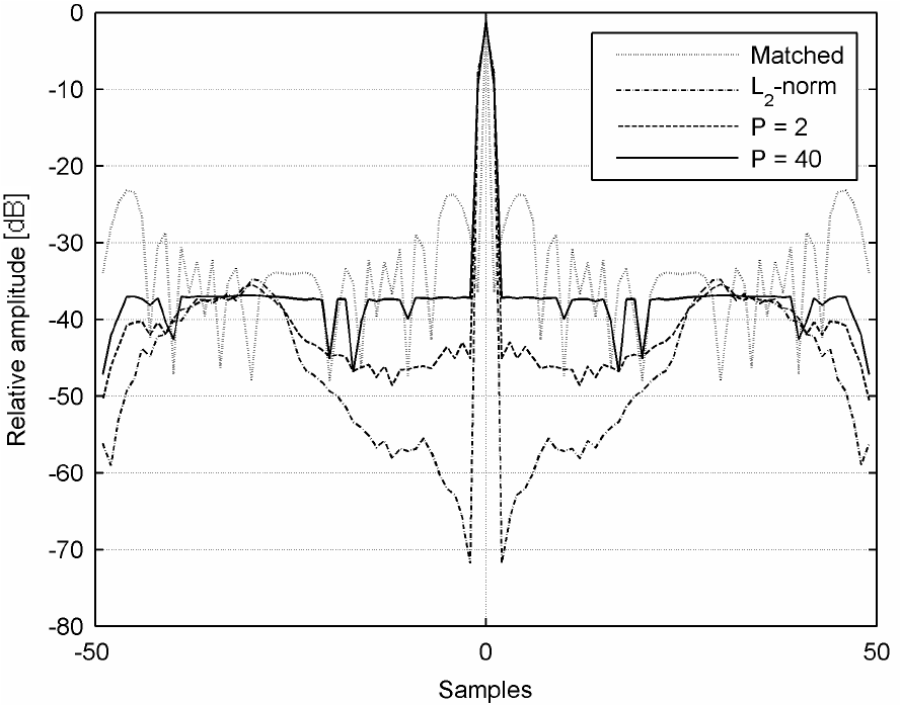


Fig. 1. Matched filter, L_2- and L_{2P}- norm pulse compression response for a linear chirp transmit pulse with a TBWP of 50 (Borrowed from [3])

4 Numerical Results

For this experiment we used a second order RKHS polynomial kernel. The MATLAB programming environment [17] was used for implementation. A linear chirp pulse

$$y(t) = Ae^{j2\pi ft}, \tag{19}$$

where A is the amplitude, f is frequency and t is time, is used to simulate the signal that should be transmitted by the radar. The generated signal had a time-bandwidth product (TBWP) of 50.

The results from [3] are shown in Fig. 1. The matched filter response, mismatched least squares and two L_{2P}-norm filter responses ($P = 2$, $P = 40$) are shown. This figure shows only the output of the compressor for the input signal that was used determine ("train") the filter coefficients. The best sidelobe levels achieved were around -35dB. The value of the output in dB is calculated as

$$\text{Output}_{dB} = 20 \log_{10} (\|\text{Output}\|) . \tag{20}$$

Fig. 2 to Fig. 4 show the results for the proposed RKHS pulse compressor. A second degree polynomial kernel was used for this RKHS method, and a chirp pulse with amplitude $A = 1$ was used to train the system. Fig. 2 shows the output of the RKHS pulse compressor for four different values of the input chirp pulse, namely

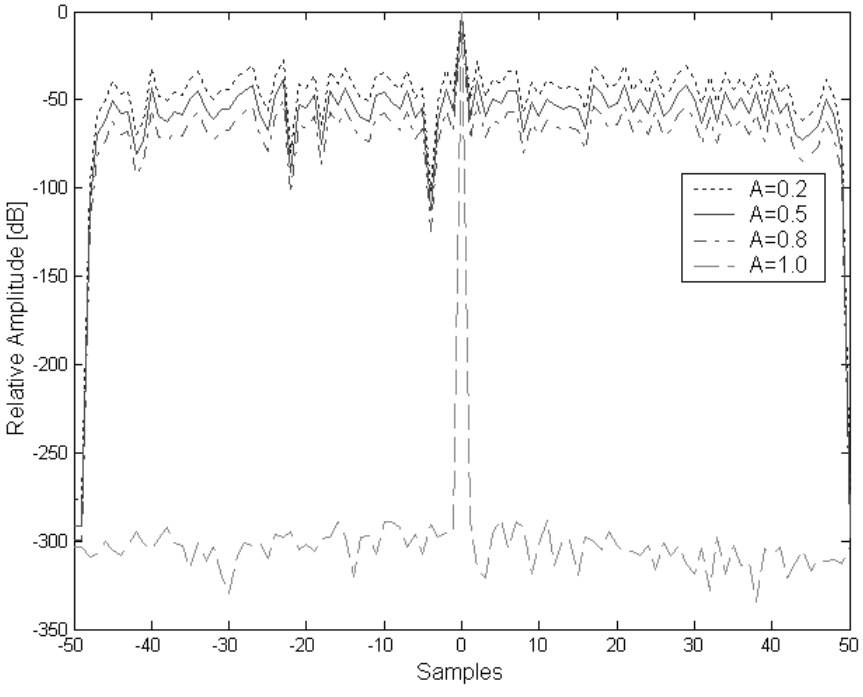


Fig. 2. RKHS pulse compressor output as the input chirp signal amplitude varies

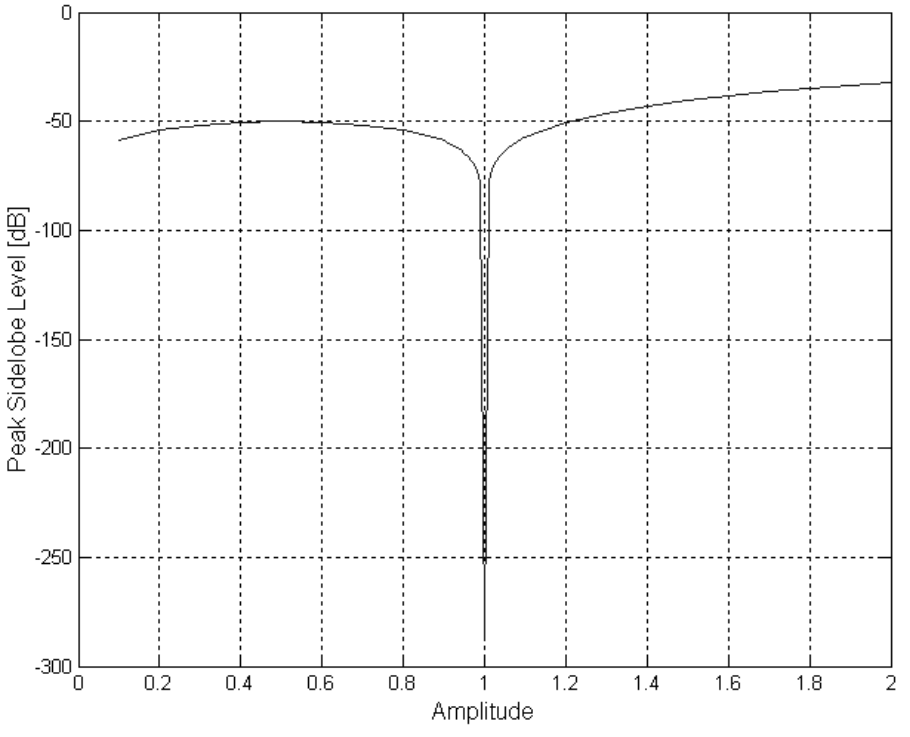


Fig. 3. RKHS maximum sidelobe levels as the input chirp signal amplitude varies

$A = 0.2$, $A = 0.5$, $A = 0.8$ and $A = 1$. Each output signal was scaled by its maximum value and then converted to the dB scale. This forces the maximum value (in dB) for each output equal to 0dB, which enables us to see the different outputs of the different input amplitudes in perspective of the output amplitude of the training chirp pulse. For the training signal with amplitude $A = 1$ it is clear that the RKHS pulse compressor performs very well compared to the least squares method, and as the amplitude deviates from 1, the sidelobe levels start to increase.

Fig. 3 shows a graph of the unscaled maximum sidelobe levels as the chirp pulse amplitude varies from $A = 0.1$ to $A = 2$. For a chirp signal with the training amplitude $A = 1$, the RKHS pulse compressor achieved sidelobe levels of maximum -287dB, which is far better than the -35dB of the pulse compressors shown in Fig. 1. Then as the input amplitude decreases or increases from that of the training pulse, the sidelobe levels varied from -60dB to -30dB. This is still better (or comparable) to the results from the L_{2P}-norm and least squares methods in Fig. 1, which only shows results for the training signal and not for input signals different than that of the training set. Fig. 4 shows the output peak amplitude (when a pulse was actually detected) in terms of the input amplitude. Since a second degree polynomial kernel was used, the output has a quadratic relation to the input amplitude.

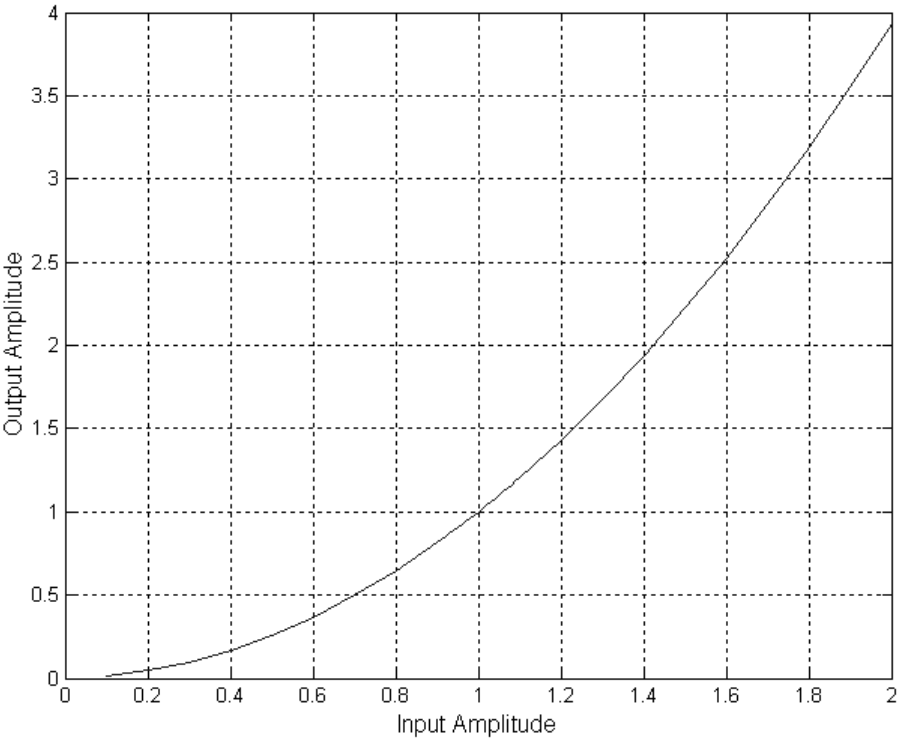


Fig. 4. RKHS pulse compressor output when a pulse is detected as the input chirp signal amplitude varies

5 Conclusion

In this paper we have presented a RKHS method to be applied in radar pulse compression. We compared our proposed method to the least squares and L_{2P}-norms for minimising pulse compression sidelobes. The RKHS method has superior performance over the other methods and showed significant sidelobe reduction of between -30dB and -287dB.

Acknowledgement

The authors would like to thank J.E. Cilliers at the Defence, Peace, Safety and Security (DPSS) operating unit of the Council for Scientific and Industrial Research (CSIR, South Africa) for his help in preparation of this paper. The authors also thank the National Research Foundation of South Africa for their financial support.

References

1. Cook, C.E.: The early history of pulse compression radar-the history of pulse compression at Sperry Gyroscope Company. IEEE Transactions on Aerospace and Electronic Systems 24, 825–833 (1988)
2. Siebert, W.M.: The early history of pulse compression radar-the development of AN/FPS-17 coded-pulse radar at Lincoln Laboratory. IEEE Transactions on Aerospace and Electronic Systems 24, 833–837 (1988)
3. Cilliers, J.E., Smit, J.C.: Pulse Compression Sidelobe Reduction by Minimization of Lp-norms. Accepted for publication in IEEE Transactions on Aerospace and Electronic Systems (2007)
4. Lewis, B.L., Kretschmer, F.F., Shelton, W.W.: Aspects of Radar Signal Processing. Artech House, Norwood, MA (1986)
5. Aronszajn, N.: Theory of Reproducing Kernels. Transactions of the American Mathematical Society 68, 337–404 (1950)
6. Parzen, E.: Statistical Inference on Time Series by Hilbert Space Methods. Technical report, Department of Statistics, Stanford University, Technical Report No. 24 (1959)
7. Parzen, E.: Regression Analysis of Continuous Parameter Time Series. In: Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability Theory, pp. 469–489. University of California Press, Berkeley, CA (1961)
8. Parzen, E.: An Approach to Time Series Analysis. The Annals of Mathematical Statistics 32, 337–404 (1961)
9. Kailath, T.: RKHS Approach to Detection and Estimation Problems—Part I: Deterministic Signals in Gaussian Noise. IEEE Transactions on Information Theory IT-17, 530–549 (1971)
10. Kailath, T., Duttweiler, D.: An RKHS Approach to Detection and Estimation Problems—Part II: Gaussian Signal Detection. IEEE Transactions on Information Theory IT-21, 15–23 (1975)
11. Kailath, T., Duttweiler, D.: An RKHS Approach to Detection and Estimation Problems—Part III: Generalized Innovations Representations and a Likelihood-Ratio Formula. IEEE Transactions on Information Theory IT-18, 730–745 (1972)
12. van Wyk, M.A., Durrani, T.S.: A Framework for Multi-Scale and Hybrid RKHS-Based Approximators. IEEE Transactions on Signal Processing 48, 3559–3568 (2000)
13. van Wyk, M.A., Durrani, T.S., van Wyk, B.J.: A RKHS Interpolator-Based Graph Matching Algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 988–995 (2002)
14. van Wyk, M.A.: Hilbert Space Methods for Non-linear Function Approximation and Filtering. Technical report, Tshwane University of Technology, CSIR / LEDGER, South Africa (2006)
15. Luenberger, D.G.: Optimization by Vector Space Methods. John Wiley and Sons, New York, NY (1969)
16. van Wyk, B.J., van Wyk, M.A., Noel, G.: Kernel-based Non-linear Template Matching. In: Fred, A., Caelli, T.M., Duin, R.P.W., Campilho, A., de Ridder, D. (eds.) SSPR 2004. LNCS, vol. 3138, pp. 831–839. Springer, Heidelberg (2004)
17. Mathworks: MATLAB Documentation - Neural Network Toolbox. Version 6.5.0.180913a Release 13 edn. Mathworks Inc., Natick, MA (2002)

Support Kernel Machine-Based Active Learning to Find Labels and a Proper Kernel Simultaneously

Yasusi Sinohara[1] and Atsuhiro Takasu[2]

[1] Central Research Institute of Electric Power Industry,
2-11-1 Iwado-kita, Komae-shi, Tokyo, 201-8511, Japan
sinohara@criepi.denken.or.jp
[2] National Institute of Informatics,
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
takasu@nii.ac.jp

Abstract. SVM-based active learning has been successfully applied when a large number of unlabeled samples are available but getting their labels is costly. However, the kernel used in SVM should be fixed properly before the active learning process. If the pre-selected kernel is inadequate for the target data, the learned SVM has poor performance. So, new active learning methods are required which effectively find an adequate kernel for the target data as well as the labels of unknown samples.

In this paper, we propose a two-phased SKM-based active learning method and a sampling strategy for the purpose. By experiments, we show that the proposed SKM-based active learning method has quick response suited to interaction with human experts and can find an appropriate kernel among linear combinations of given multiple kernels. We also show that with the proposed sampling strategy, it converges earlier to the proper combination of kernels than with the popular sampling strategy MARGIN.

1 Introduction

Active learning are used when a large number of unlabeled samples are available but getting their labels is costly, usually in cases that human experts assess the labels of unlabeled samples. Support vector machine (SVM)-based active learning has been successfully applied but the kernel used in SVM should be fixed properly in advance. If the pre-selected kernel is inadequate for the target data, the learned SVM has low predictive power. In batch learning, we can use time-consuming cross validation or other methods to find a proper kernel. But in active learning interacting with a human expert, the turnaround time, i.e., the time it takes to show an unlabeled sample for next labeling after one labeling, should be kept short. So a quickly responding active learning method is necessary which effectively finds an adequate kernel for the target data as well as the labels of unknown samples by interacting with experts.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 277–286, 2007.

In this paper, we propose a support kernel machine (SKM)-based active learner for the purpose. Because solving SKM is more time-consuming than SVM, we propose a two-phased SKM solver to reduce the turnaround time and also propose a sampling strategy SKM-SHIFT for SKM-based active learning.

2 Support Kernel Machines

Both SVM[1] and SKM[2] learn a separator $f(x)$ and predict the label $y \in \{\pm 1\}$ of input x by $y = sign(f(x))$.

For a given set of training samples $\{(x_i, y_i)\}_{i=1,\ldots,N}$ and a given (non-linear) feature mapping $\phi(x)$ or kernel $K(x,z) = \phi(x)^T \phi(z)$, SVM finds a large margin separator $f(x) = w^T\phi(x) + b$ by solving the following optimization problem.

$$\min_{\substack{w,b \\ \xi_i \geq 0}} \tfrac{1}{2}||w||^2 + C\textstyle\sum_i \xi_i \;\text{ subject to } y_i(w^T\,\phi(x_i) + b) \geq 1 - \xi_i \tag{1}$$

The first term is for maximization of the margin $1/||w||$ and the second term is for the minimization of errors and the cost parameter C controls the trade-off between them. We shall henceforth refer to the optimal objective function value (1) as the error index which is closely related to the generalization error [1].

By duality, the following $\max_\alpha S(\alpha)$ equals to the error index.

$$\max_{\substack{\alpha_i \in [A_i, B_i] \\ \sum \alpha_i = 0}} \underbrace{\sum_i \alpha_i y_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j K(x_i, x_j)}_{S(\alpha)} \tag{2}$$

$$\text{where } \;[A_i, B_i] = [\min(y_i\, C, 0), \max(y_i\, C, 0)]$$

By the optimal α^*, the optimal $w = \sum \alpha_i^*\, \phi(x_i)$ and $f(x) = \sum \alpha_i^* K(x_i, x) + b$. The samples whose $\alpha_i^* \neq 0$ are called support vectors.

In contrast to the SVM which uses a given single kernel $K(x_i, x_j)$, the SKM searches the SVM with the least error index whose kernel is a linear combination of given M kernels $\sum_{k=1}^M \beta_k\, K_k(x_i, x_j)$. Therefore SKM solves the following min-max or dual max-min problem [2].

$$\min_{\substack{\beta_k \geq 0 \\ \sum \beta_k = 1}} \max_{\substack{\alpha_i \in [A_i, B_i] \\ \sum \alpha_i = 0}} S(\alpha; \beta) \tag{3}$$

$$= \max_{\substack{\alpha_i \in [A_i, B_i] \\ \sum \alpha_i = 0}} \min_{\substack{\beta_k \geq 0 \\ \sum \beta_k = 1}} S(\alpha; \beta) \tag{4}$$

$$\text{where } S(\alpha;\beta) = \textstyle\sum_i \alpha_i y_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j \underbrace{\left(\sum_k \beta_k K_k(x_i, x_j)\right)}_{K(x_i, x_j; \beta)}$$

$$= \textstyle\sum_k \beta_k \underbrace{\left(\sum_i \alpha_i y_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j K_k(x_i, x_j)\right)}_{S_k(\alpha)}$$

We refer $K(x_i, x_j; \beta)$ as a composite kernel and $K_k(x_i, x_j)$ as the k-th component kernel. $S_k(\alpha)$ is the objective function of the SVM with k-th component kernel.

SKM problem (3),(4) is an SVM problem w.r.t. α and a linear programming (LP) w.r.t. β, so the optimal α^*,β^* are sparse. The samples whose $\alpha_i^* \neq 0$ and the kernels whose $\beta_k^* > 0$ are called the support vectors and the support kernels respectively. The optimal $w = [\beta_1^* w_1, \cdots, \beta_M^* w_M]^\top$ where $w_k = \sum_i \alpha_i^* \phi_k(x_i)$ and $f(x) = \sum_k \beta_k^* \sum_i \alpha_i^* K_k(x_i, x) + b$. Thus, SKM can extract only critical samples and kernels for classification from given samples and component kernels.

SKM's error index $S(\alpha^*; \beta^*)$ equals to all support kernels' $S_k(\alpha^*)$ [1] and is smaller than any single kernel SVM's error index $\max_\alpha S_k(\alpha)$ because only weak duality ($\max_\alpha \min_k S_k(\alpha) \leq \min_k \max_\alpha S_k(\alpha)$) holds. Consequently SKM is expected to have higher precision than any single kernel SVM.

3 Two-Phased SKM-Based Active Learning: LASKM

SKM problem (3) is equivalent to the following semi-infinite LP (SKM-ILP).

$$\min_{\substack{\theta, \beta \geq 0 \\ \sum_k \beta_k = 1}} \theta \text{ s.t. } \sum_k \beta_k S_k(\alpha) \leq \theta \text{ for all } \alpha \in \{\alpha | \alpha_i \in [A_i, B_i], \textstyle\sum_i \alpha_i = 0\} \quad (5)$$

SKM-ILP can be solved by repeating the following steps starting from the initial constraint set $\mathcal{CS} = \{\beta_k \geq 0, \sum_k \beta_k = 1\}$ and some α^0,β^0 [3].

S1. find α^t s.t. $S(\alpha^t; \beta^{t-1}) > S(\alpha^{t-1}; \beta^{t-1})$.
S2. add constraint $\sum_k \beta_k S_k(\alpha^t) \leq \theta$ to the constraint set $\mathcal{CS}$ of LP.
S3. get β^t by solving the LP problem $\min_{(\theta,\beta)\in\mathcal{CS}} \theta$.

$\alpha^t = \operatorname{argmax} S(\alpha; \beta^{t-1})$, the solution of SVM with kernel $K(\cdot, \cdot; \beta^{t-1})$, gives the tightest constraint at β^{t-1} in S2. However solving SVM is time-consuming and moreover it takes M times more time to solve $\max S(\alpha; \beta^{t-1})$ because the composite kernel has M component kernels. So the reduction of computation time of SVM is important to keep quick response during the active learning.

We therefore take a two-phased approach. During the active learning phase, we partially solve the SVM quickly using LASVM [4] and in the post-optimization phase, we completely optimize the SVM for all labeled data using a normal solver. The proposed two-phase active learner LASKM is shown in Algorithm 1.

LASVM is an efficient online SVM algorithm competitive with misclassification rates and out-speeding state-of-the-art SVM solvers. It is convenient to get α^t quickly in the active learning phase because it keep the set $\mathcal{S}$ of candidates of support vectors small by discarding blatant non-support vectors. It maintains three pieces of information: the set $\mathcal{S}$ and the coefficients α_i and the gradients $g_i = \frac{\partial}{\partial \alpha_i} S(\alpha) = y_i - \sum \alpha_j K(x_j, x_i)$ for $i \in \mathcal{S}$. Its building block operations are PROCESS and REPROCESS. $PROCESS(i)$ attempts to insert sample i into

[1] $S(\alpha^*; \beta^*) = \max_\alpha \min_k S_k(\alpha) = \min_k S_k(\alpha^*)$ because the inside of (4) is the LP-formulation of $\min_k S_k(\alpha)$. β^* can be determined because it satisfies $\sum \beta^* S_k(\alpha) \leq S(\alpha^*; \beta^*)$ for all α.

$\mathcal{S}$ and updates α_i and g_i for $i \in \mathcal{S}$. $REPROCESS$ removes blatant non-support vectors from $\mathcal{S}$ and updates α_i, g_i for $i \in \mathcal{S}$.

In LASKM, we extend PROCESS and REPROCESS to $PROCESS(i, \beta)$ and $REPROCESS(\beta)$ which maintain the gradients of each component kernel $g_{[k]i} = y_i - \sum_j \alpha_j K_k(x_j, x_i)$ as well as that of the composite kernel g_i.

Algorithm 1. Two-Phased SKM-based Active Learner: LASKM

Require: Cost parameter C and Component kernels $K_k(\cdot,\cdot)$, $k = 1, \cdots, M$
Require: samples $(x_i, y_i), i = 1, \cdots, N$ and Sampling Strategy SS
//Initialization
$t \leftarrow 0$.
constraint set $\mathcal{CS} \leftarrow \{\beta_k \geq 0, \sum \beta_k = 1\}$.
initial $(\alpha^0, g^0, g^0_{[k]}) \leftarrow (0, y, y)$, $\beta^0 \leftarrow (1/M, \cdots, 1/M)$.
Seed L with a few samples of each class.
update $(\alpha^t, g^t, g^t_{[k]})$
//Active Learning Phase
repeat
  $t \leftarrow t + 1$
  Pick a sample i_t by sampling strategy SS and $L \leftarrow L \cup \{i_t\}$.
  update $(\alpha^t, g^t, g^t_{[k]})$ by PROCESS(i_t, β^{t-1}) and REPROCESS(β^{t-1}).
  $S^t_k \leftarrow S_k(\alpha^t), k = 1, \cdots, M$
  $CS \leftarrow CS \cup \{\sum_k \beta_k S^t_k \leq \theta\}$. //remove redundant constraints
  $(\theta^t, \beta^t) \leftarrow \mathrm{argmin}\{\theta | (\theta, \beta) \in CS\}$. //LP solution $\theta^t = S(\alpha^t; \beta^t)$.
  $g \leftarrow \sum_k \beta^t g^t_{[k]}$.
until $|L| \geq N$
//Post-Optimization Phase
repeat
  $t \leftarrow t + 1$
  get (α^t, g^t, g^t_k) by solving $\max S(\alpha; \beta^{t-1})$ using normal SVM algorithm.
  $S^t_k \leftarrow S_k(\alpha^t), k = 1, \cdots, M$ and $\theta^t_0 \leftarrow \sum \beta^{t-1}_k S^t_k$. //$\theta^t_0 = S(\alpha^t; \beta^{t-1})$
  $CS \leftarrow CS \cup \{\sum_k \beta_k S^t_k \leq \theta\}$. //remove redundant constraints
  $(\theta^t, \beta^t) \leftarrow \mathrm{argmin}\{\theta | (\theta, \beta) \in CS\}$. //LP solution $\theta^t = S(\alpha^t; \beta^t)$
  $g \leftarrow \sum_k \beta^t g^t_k$.
until $\theta^t_0 > 0$ and $|1 - \theta^t / \theta^t_0| \leq \epsilon$

4 Sampling Strategies for SKM-Based Active Learning

The typical sampling strategies for SVM-based active learning are the followings:

1. RANDOM selects a sample randomly
2. MARGIN selects the sample nearest to the boundary ($\mathrm{argmin}_x |f(x)|$).
3. KFF (Kernel Farthest First) selects the sample farthest to the boundary ($\mathrm{argmax}_x |f(x)|$).
4. SHIFT selects the sample initially by KFF and by MARGIN depending on the stability of the model.

MARGIN strategy is known to be the most effective when the current model (α) is near optimal. However, when the current model is far from the optimal, true support vectors can exist far from the current boundary and exploration is more important than exploitation of the current boundary. SHIFT[5] initially uses KFF for exploration and shifts to MARGIN when the model becomes stable.

In case of SVM-based active learning, SHIFT's improved performance was limited on data sets that require extensive exploration such as checkerboard or COREL dataset, while remaining competitive on data sets that do not[5]. But SKM's model space (α, β) is larger than SVM's. So in SKM-based active learning, we think balancing exploration and exploitation is more important and propose to use an extended SHIFT strategy for the SKM-based active learning. The algorithm is shown in Algorithm 2. Along the line in [5], we defined the instability of SKM by the instability of angles between $w^t = (\beta_1 w_1^t, \cdots, \beta_M w_M^t)$ and w^{t-1}.

$$\mathrm{corr}(\alpha^t, \beta^t, \alpha^{t-1}, \beta^{t-1}) = \frac{\langle w^t, w^{t-1} \rangle}{\sqrt{\langle w^t, w^t \rangle \langle w^{t-1}, w^{t-1} \rangle}}$$

$$\text{where } \langle w^u, w^v \rangle = \sum_{i,j,k} \beta_k^u \beta_k^v \alpha_i^u \alpha_j^v K_k(x_i, x_j)$$

Algorithm 2. SKM-SHIFT(α, β, U)

Require: U:unlabeled samples, O:oracle
Require: λ, ϵ, A:learning algorithm
if $U \neq \{\}$ **return ({}, U) end if**
$\phi \leftarrow \mathrm{corr}(\alpha, \beta, \alpha^0, \beta^0)$.
$p \leftarrow \max(\min(p_0 e^{-\lambda(\phi - \phi_0)}, 1 - \epsilon), \epsilon)$.
// probabilistic switching of strategy
$r \leftarrow$ random number generated uniformly between 0 and 1.
if $r < p$ **then**
  $x \leftarrow \mathrm{argmax}_{x \in U} \min_{i \in \mathcal{S}} K(x_i, x)$ //exploration: KFF(U)
else
  $x \leftarrow \mathrm{argmin}_{x \in U} |f(x)|$. //exploitation:MARGIN$(U, \alpha, \beta)$
end if
$y \leftarrow O(x)$. //get the label of x from the oracle
$(p_0, \alpha^0, \beta^0) \leftarrow (p, \alpha, \beta)$. //update state variables
return $((x, y), U - \{x\})$

When there are many unlabeled samples and we use all unlabeled samples as U in SKM-SHIFT/MARGIN/KFF, it takes much time to evaluate $|f(x)|$ or $K(x_i, x)$ of all x in U. To keep the turnaround short, we use a fixed number of randomly selected unlabeled samples as U. We set $|U| = 50$. This setting is practical because the probability that the maximum in U is over the 95th or 90th percentile of all is about 92% $(= 1 - 0.95^{50})$ or 99.5% $(= 1 - 0.90^{50})$.

MARGIN and KFF need the evaluation time proportional to $M \times |\mathcal{S}|$ to compute kernels between each sample in U and each sample in $\mathcal{S}$.

SKM-SHIFT needs additionally to compute $\mathrm{corr}(\alpha^t, \beta^t, \alpha^{t-1}, \beta^{t-1})$. However, because LASKM stores $\langle w^{t-1}, w^{t-1} \rangle$ and $g^t_{[k]j} = y_j - \sum_i \alpha^t_i K_k(x_i, x_j)$ in memory and $\langle w^t, w^s \rangle = \sum_{j,k} \alpha^s_j \beta^t_k \beta^s_k \left(\sum_i \alpha^t_i K_k(x_i, x_j)\right)$, SKM-SHIFT needs to calculate $\sum_i \alpha^t_i K_k(x_i, x_j)$ only for the samples j which are the support vector candidates at time $t-1$ but excluded at time t (usually a few at most). So, SKM-SHIFT can select samples in almost equivalent time to MARGIN's.

5 Experiments

In the experiments, we use the USPS database [2] which contains 9298 handwritten digits (7329 for training, 1969 for testing). Each digit is a 16×16 image with zero mean and variance 1. We prepared 6 RBF kernels $\exp(-\gamma||x_i - x_j||^2)$ with $\gamma = 1, 2, 5, 10, 20, 50$ and make each single kernel SVM and the SKM using all 6 kernels learn the training data in batch or active learning. In batch learning, we use libsvm [6] for SVM and post-optimization part of LASKM for SKM. In active learning, we use LASVM and LASKM for SVM and SKM respectively. We set the cost parameter $C = 1000$ assuming the dataset is nearly separable, and tolerance $\tau = 1^{-3}$ (libsvm's default) for both SVM and SKM and the upper bound of $1 - \theta^t/\theta^t_0 = 1^{-6}$ for SKM. We use RANDOM, MARGIN, SKM-SHIFT as sampling strategies. KFF is excluded because it is obviously ineffective. The parameters in SKM-SHIFT(Algorithm 2.) are $\lambda = 0.5$, $\phi_0 = 0.3$, $\epsilon = 0.05$ [3] and the size of random sampling is $|U| = 50$.

Usually assessing labels of over 1000 samples are very stressful. So we set the maximum number of learned samples to 900 in active learning. The analysis of changes of turnaround time, precision etc. are based on the data when the numbers of learned samples are 12, 14, ..., 28, 30, 40, 50, ..., 90, 100, 200,..., 800, 900.

We made 20 trials for each experimental setting. In each trial, we randomly select a pair of one positive and one negative sample as the initial sampling data.

We use a Pentium-4 2.6GHz Windows XP machine for the experiments.

5.1 Comparison Between Single Kernel SVMs and SKM

We first compared the optimal single kernel SVMs and the optimal SKM when learning all training data in batch.

Table 1 lists the error index ($\max_\alpha S_k(\alpha)$ of SVMs and $S(\alpha^*; \beta^*)$ of SKM) and precision against the test data for digit '0'.

SKM's error index is the lowest among all and the optimal weights β^*_γ of the support kernels are 0.5 (γ=5), 0.2 (γ=10), 0.3 (γ=20) and those of the non-support kernels (γ=1, 50) are 0. SKM successfully selected the three kernels with low error index as the support kernels and composed a composite kernel with a lower error index. The precision of the optimal SKM is 99.54% which is the

[2] ftp://ftp.kyb.tuebingen.mpg.de/pub/bs/data/

[3] λ, ϕ_0,ϵ are not optimized. we set them referring to [5].

Table 1. Error Index and Precision for '0'

γ	Error Index	Precision
1	1334.1	99.44%
2	467.6	99.49%
5	175.6	99.54%
10	163.3	99.64%
20	331.4	98.93%
50	1082.4	93.30%
SKM	149.8	99.54%

Table 2. β^* selected by SKM

digit	RBF parameterγ					
	1	2	5	10	20	50
0	0.00	0.00	0.52	0.19	0.29	0.00
1	0.00	0.00	0.48	0.24	0.22	0.06
2	0.00	0.00	0.48	0.28	0.13	0.11
3	0.00	0.00	0.53	0.17	0.29	0.01
4	0.00	0.00	0.44	0.36	0.11	0.09
5	0.00	0.00	0.46	0.29	0.15	0.10
6	0.00	0.00	0.47	0.21	0.32	0.00
7	0.00	0.00	0.52	0.20	0.24	0.04
8	0.00	0.00	0.44	0.37	0.03	0.16
9	0.00	0.00	0.42	0.38	0.16	0.05

second best behind the SVM's with $\gamma = 10$ (99.64%) and the SMV's with $\gamma = 50$ is the worst (93.30%). For other digits, SKMs also had the least error index and the best or second best precision compared with the single kernel SVMs [4].

In active learning, this feature is a great advantage of SKM against SVM. Because the user doesn't know the labels of data when selecting kernels, there are relatively high risks that the user selects an inefficient kernel (such as $\gamma = 50$ for digit '0') in SVM-based active learning but in SKM-based active learning, we can reach to the solution comparable to the best single kernel SVM's starting from a set of candidate kernels.

Table. 2 shows the optimal β^* of SKM for each digit. For all digits, the SKM's major support kernels are γ =5, 10, 20 and γ =5 has the highest weight. However, the SVM with the least error index among the single kernel SVMs is γ =10 and not γ =5. In the optimal SKM, the RBF kernel with $\gamma = 5$ gives a rough shape of the decision boundary and the RBFs with $\gamma = 10$ or 20 refine it.

5.2 Comparison of Sampling Strategy in SKM-Based Active Learning

Turnaround Time. Figure. 1 shows the average turnaround time (dotted line) and the average number of support vector candidates $|\mathcal{S}|$ (solid line). The numbers of learned samples are on the horizontal axis.

RANDOM responses very rapidly (0.02 sec. at 900 samples) but it fails to select proper support vector candidates effectively. So the number of candidates stays 200, the one third of those of the other two strategies at 900 samples.

MARGIN and SHIFT have similar turnaround proportional to the number of support vector candidates. Even at 900 samples, it responses within one second (0.9 sec.).

As for the simple active SKM learner, which uses MARGIN and incrementally optimizes SKM completely at each sampling, its average turnaround was 2.5 sec.

[4] The error index is only an "estimate" of generalization error. So the precision of the optimal SKM is comparable to the best SVM but can be inferior to it in some cases.

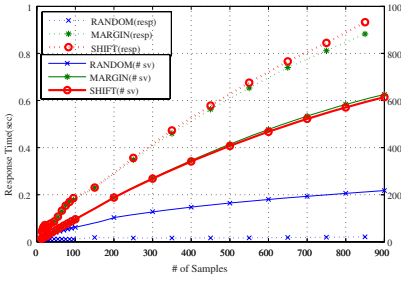


Fig. 1. Turnaround time and the number of SVs of LASKM

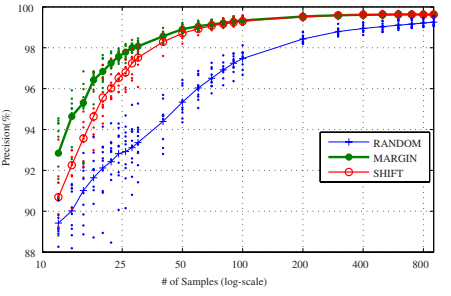


Fig. 2. Transition of Precision

for 100–200 samples, 10 sec. for 200–300 samples, 15 sec. for 300–400 samples. These responses are too slow for comfortable interaction with human experts. In contrast, the two-phased active learner LASKM has very quick response suitable to support interactive active learning.

Temporal Transition of Precision. We show the changes of precision averaged over 200 trials (20 trials of ten digits) for each strategy in Figure 2 because the performance are very similar regardless of digits.

RANDOM is obviously poor and MARGIN and SKM-SHIFT have similar performance in precision. More precisely, MARGIN is slightly better than SKM-SHIFT when the number of learned samples is small (less than 200 for digit '3' and '4' and 30 to 80 for other digits) but the differences of precision between MARGIN and SKM-SHIFT are less than 0.5% even at 50 samples for all digits.

Concerning precision, SKM-SHIFT didn't make much difference with MARGIN just like the SHIFT in the SVM-active learning as described in [5].

Temporal Transition of Kernel Weight. Concerning the changes of kernel weight β^t, the performances of MARGIN and SKM-SHIFT are different.

Figure 3 shows the temporal changes of kernel weights β^t (averaged over 20 trials) of digit '0'. The optimal $\beta^* = (0, 0.5, 0.2, 0.3, 0)$ is shown in the right most in Figure3. MARGIN gives more weight to $\gamma = 10$ until the number of samples reaches about 200 and requires over 400 samples to reach β^*. On the other hand, SHIFT reached β^* at about 200 samples.

The same tendency appears in other digits. MARGIN is likely to select $\gamma = 10$ which has the least error index in single kernel SVMs in the earlier stage ($\beta_{\gamma=10}$ is around 0.5 or 0.6 at 20 samples). SHIFT is likely to reach to β^* earlier than MARGIN.

Figure 4 shows the changes of $\|\beta^t - \beta^*\|$, the estimation error of β^*. Especially, SHIFT's β^t are closer to β^* than MARGIN until 100 samples except for digits '4', '8' and '9' (at significance level 90%).

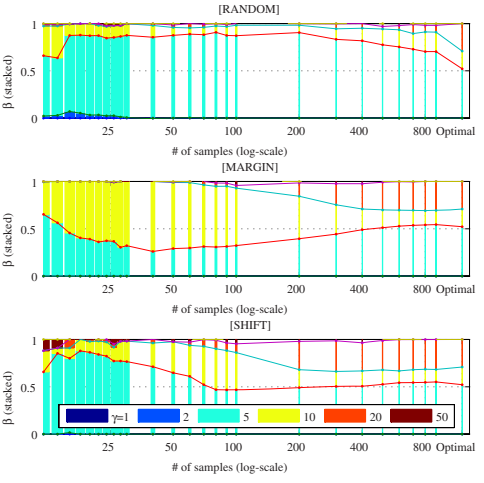


Fig. 3. Transition of β

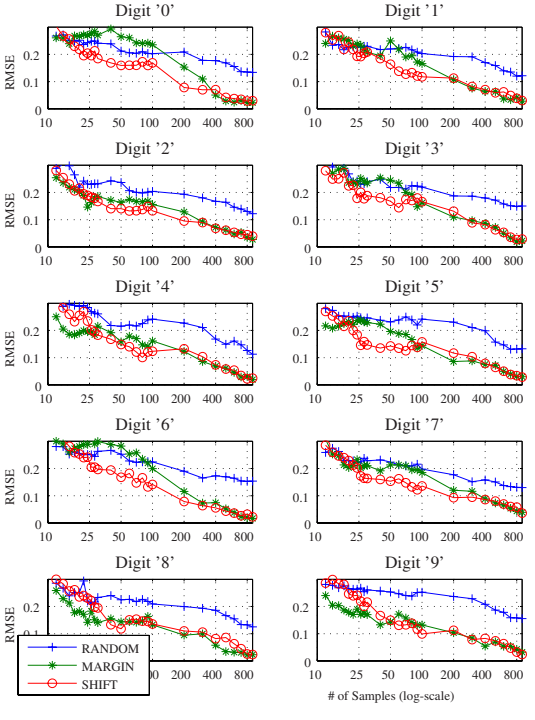


Fig. 4. Transition of Errors of β, $||\beta - \beta^*||$

The reason of the exceptional good performance of MARGIN for digit '4', '8', '9' can be considered as follows. MARGIN is likely to select $\gamma = 10$ in initial phase as mentioned above and digits '4', '8', '9' have larger optimal weight $\beta^*_{\gamma=10}$ than other digits as shown in Table 2. So, the initial decision boundary by MARGIN is likely to be closer to the optimal boundary and this causes the good estimate of β^* at earlier stages by MARGIN.

But in general, the best kernel for single kernel SVM is different from the kernel having large weight in SKM. So, SKM-SHIFT can be considered as more robust and better estimator of optimal kernel weight β^* than MARGIN.

6 Conclusions and Future Works

In this paper, we propose the SKM-based active learning and, for the purpose, a two-phased algorithm LASKM and a sampling strategy SKM-SHIFT based on SHFIT strategy whose improvement was limited in case of SVM-based learning.

By experiments, we show that the proposed LASKM has quick response necessary for interactive active learning and can find an appropriate composite kernel among combinations of given component kernels which is comparable to the best component kernel with respect to the predictive power.

We also show that with the proposed sampling strategy SKM-SHIFT it converges earlier to the optimal combination β^* of composite kernel than with the popular sampling strategy MARGIN with some exception, while it remains comparable with MARGIN concerning precision of prediction of labels.

We are now conducting empirical evaluations of the LASKM for different datasets and are also planning detailed sensitivity analyses of SKM-shift's parameters.

References

1. Vapnik, V.: Statistical Learning Theory. John Wiley and Sons Inc., Chichester (1998)
2. Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple Kernel Learning, Conic Duality, and the SMO Algorithm. In: Proc. ICML, Banff Canada (2004)
3. Sonnenburg, S., Ratsch, G., Schafer, C.: A general and efficient multiple kernel learning algorithm. In: Advances in Neural Information Processing Systems, vol. 18, MIT Press, Cambridge MA (2006)
4. Bordes, A., Ertekin, S., Weston, J., Bottou, L.: Fast Kernel Classifiers with Online and Active Learning. Journal of Machine Learning Research 6, 1579–161 (2005)
5. Osugi, T., Kun, D., Scott, S.: Balancing Exploration and Exploitation: A New Algorithm for Active Machine Learning. In: Proc. ICDM 2005, pp. 330–337 (2005)
6. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001), Software available at http://www.csie.ntu.edu.tw/~emcjlin/libsvm

Making Class Bias Useful: A Strategy of Learning from Imbalanced Data

Jie Gu[1], Yuanbing Zhou[2], and Xianqiang Zuo[2]

[1] Software School of Tsinghua University
guj05@mails.tsinghua.edu.cn
[2] State Power Economic Research Institute, China
{zhouyuanbing,zuoxinqiang}@chinasperi.com.cn

Abstract. The performance of many learning methods are usually influenced by the class imbalance problem, where the training data is dominated by the instances belonging to one class. In this paper, we propose a novel method which combines random forest based techniques and sampling methods for effectively learning from imbalanced data. Our method is mainly composed of two phases: data cleaning and classification based on random forest. Firstly, the training data is cleaned through the elimination of dangerous negative instances. The data cleaning process is supervised by a negative biased random forest, where the negative instances have a major proportion of the training data in each of the tree in the forest. Secondly, we develop a variant of random forest in which each tree is biased towards the positive class to classify the data set, where a major vote is provided for prediction. In the experimental test, we compared our method with other existing methods on the real data sets, and the results demonstrate the significative performance improvement of our method in terms of the area under the ROC curve(AUC).

1 Introduction

In recent years, learning from imbalanced data has received increasing interest from the community of machine learning and data mining. Imbalanced data sets exhibit skewed class distributions in which almost all instances are belonging to one or more larger classes and far fewer instances belonging to a smaller, but usually more interesting class. This kind of data can be found in many real-world applications, such as power load data, medical data, sales data, etc.

The balance of training set is an underlying assumption for most learning systems, thus the performance of these systems can be influenced greatly when learn from imbalanced data. More specifically, models trained from imbalanced data sets are biased towards the majority classes and intended to ignore the minority but interested classes. This is the class bias problem, which can lead to a poor performance of the classifier when deal with the class that is not biased. As an example, for a data set where only 1% instances are positive, an accuracy of 99% will be achieved simply by classifying all the instances to be negative. Such is a typical classifier with high accuracy but useless when exploited to classify the positive instances.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 287–295, 2007.

Class bias is the major reason of the decline of classification performance on imbalanced data, and thus it is by no means trivial to explore a solution to address the problem. In this paper, we propose a strategy of learning from the imbalanced data set, where class bias is utilized to help improving the classification performance in a proper way. Through analyzing the issue in detail, we can conclude that class bias is not the only reason for the loss of performance, and class overlapping is another problem that could also hinder the performance. In our method, firstly, class bias is exploited to settle class overlapping in the data cleaning process. Secondly, we provide a combination of ensemble learning methods and sampling techniques, where the final prediction is made based on biased classifiers. In the experimental evaluation, we compared our method with other existing methods on the real data sets, and the results demonstrate superior performance of the proposed methods.

The remainder of this paper is organized as follows. In Section 2, we give a brief review of the related work. Our method is described in detail in Section 3. Section 4 gives the experimental evaluation on competitive methods. Finally, Section 5 offers the conclusion remarks.

2 Related Work

A number of solutions to the class-imbalance problem were previously proposed both at the data and algorithmic levels [2]. At the data level, these solutions are based on many different forms of re-sampling techniques including under-sampling the majority and over-sampling the minority to balance the class distribution. At the algorithmic level, the frequently used methods includes cost-sensitive classification [3], recognition-based learning [4], etc.

Random forest [1] is an ensemble of unpruned classification or regression trees, trained from bootstrap samples of the training data, using random feature selection in the tree induction process. The classification is made through a majority vote which takes all the decision of the trees into consideration. Random forest shows important performance improvement over the single tree classifiers and many other machine learning techniques. However, random forest also suffers from the class imbalance when learning from the data set. In this paper, we made significant modification to the basic random forest algorithm to tackle the problem of learning form imbalanced data set.

3 Proposed Solutions

The key idea of our method is to utilize the class bias for removing the data, which seriously puzzle classification process, defined as “dangerous”. Our method mainly consists of two phases:

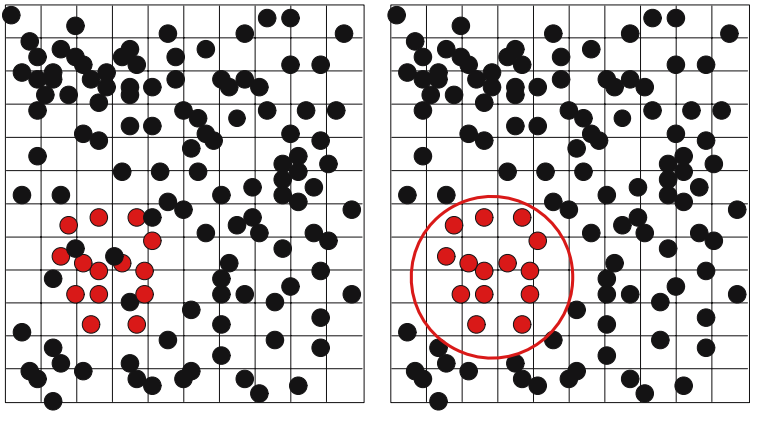
1. Data cleaning: The majority of data are preprocessed to eliminate the instances, which may cause degradation of prediction performance.

2. Classification (ranking): We build the model to produce a score for each of the instances in the given prediction data set to indicate their possibility of being positive or negative.

3.1 Data Cleaning

The degradation of performance in many standard classifiers is not only due to the imbalance of class distribution, but also the class overlapping caused by class imbalance. To better understand this problem, imagine the situation illustrated in **Figure 1**. Figure 1(a) represents the original data set without any preprocess. The circle in red and black represent the positive class and the negative class respectively, where an obvious class imbalance exists. Note that in Figure 1(a) there are several negative instances in the region dominated by the positive instances, which presents some degree of class overlapping. These negative instances are considered to be 'dangerous' since it is quite possible for any model trained from this data set to misclassify many positive instances as negative. For cost-sensitive related problems, this issue is even more detrimental. Then a natural requirement is to eliminate the dangerous negative instances, i.e., the black circles in the red region in Figure 1(a). The data set after cleaning, where a well-defined class boundary exists, is represented in Figure 1(b).

There are many existing techniques designed for this data cleaning task, including **Tomek links** [7], **Condensed Nearest Neighbor Rule** [6], etc. The main defect of these methods lies in their strong dependency on a distance function defined for the data set. However, the most effective form of the distance function can only be expressed in the context of a particular data domain. It is also often a challenging and non-trivial task to find the most effective form of



(a) Class overlapping exists before Data Cleaning

(b) Well-defined class boundary emerges after Data Cleaning

Fig. 1. Effect of Data Cleaning

the distance function for a particular data set [8]. Without a well-approximately defined distance function, the result of the data cleaning process is often very poor and cannot eliminate the dangerous negative instances effectively.

The data cleaning method proposed in this paper does not employ the utilization of any distance functions and is more straightforward. In this method, three steps are designed to implement cleaning process as follows:

1. Divide: The data are divided into the minor instances set $\mathcal{P}$ and the major instances set $\mathcal{N}$, and $\mathcal{N}$ is further divided into n subsets $\{\mathcal{N}_1, \mathcal{N}_1, ..., \mathcal{N}_n\}$, where each of them has approximately the same size.
2. Train: For each $\mathcal{N}_i$, we train a random forest RF from the rest instances in $\mathcal{N}$ and the entire $\mathcal{P}$. The trick is that for every classification tree in the forest, the class distribution in the corresponding training data is not balanced, i.e., more negative instances than positive instances.
3. Filter: We remove all instances in $\mathcal{N}_i$, which are are incorrectly classified by RF, from the training data set.

The rationale behind this data cleaning method is quite legible. Most standard learning algorithms assume that maximizing accuracy on a full range of cases is the goal and, therefore, these systems exhibit accurate prediction for the

Algorithm 1. Data Cleaning: Eliminate Dangerous Negative Instances

Input: majority instance set $\mathcal{N}$, minority instance set $\mathcal{P}$, number of subset n, number of trees in the forest l_{tree}, threshold $\epsilon \in (0, 1)$

Output:clean data set $D^{'}$ where dangerous negative instances are eliminated from D according to the given parameter ϵ

Divide $\mathcal{N}$ randomly into n subsets $\{\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_n\}$, $\forall i, j \in [1, 2, ..., n]$, $|\mathcal{N}_i| = |\mathcal{N}_j|$, $\mathcal{N}_i \cap \mathcal{N}_j = \emptyset$ if $i \neq j$.
$i \leftarrow 0$
repeat
  $i \leftarrow i + 1$
  $\mathcal{N}^{'} = \bigcup_{j=1}^{n} \mathcal{N}_j - \mathcal{N}_i$
  for $m = 1$ to l_{tree} **do**
    Randomly sample a subset $\mathcal{P}_{sub}$ from $\mathcal{P}$ and a subset $\mathcal{N}_{sub}$ from $\mathcal{N}^{'}$, where $|\mathcal{P}_{sub}| \ll |\mathcal{N}_{sub}|$
    Learning a classification tree T_m form $\mathcal{N}_{sub} \cup \mathcal{P}_{sub}$
  end for
  for each instance $X_n \in \mathcal{N}_i$ **do**
    if more than $\epsilon \times l_{tree}$ tree classifiers classify X_n as positive **then**
      Eliminate X_n from $\mathcal{N}_i$
    end if
  end for
until $i = n$
$\mathcal{N}_{out} = \bigcup_{j=1}^{n} \mathcal{N}_j$
Return $\mathcal{N}_{out}$ as the output

majority class cases, but very poor performance for minority. For such negative-biased classification model, if a negative instance is misclassified as positive, it is reasonable to consider it as dangerous since it must be highly similar with some certain positive instances and thus is responsible for class overlapping. Elimination of these dangerous negative instances from the training data will potentially reduce the false negative rate of the model trained from it. Details of this method is described in Algorithm.1.

3.2 Classification and Ranking

In this subsection, we introduce the method developed to rank the given instances in the testing data set to indicate their possibility of being positive. The basic model is similar with the random forest used in the data cleaning process. The only difference is that the class distribution of the training data for each tree in the forest has been reversed, i.e., more positive instances than negative instances. Several random forests are obtained in this way, and the instances in the prediction data set will be classified by the trained random forests in an iterative process. For each instance, the number of trees in the forest that classifies it as positive are assigned as its score, which represents the possibility of that

Algorithm 2. Classification and Ranking

Input: cleaned training data set $\mathcal{D}_t$, prediction data set $\mathcal{D}_p$, number of iterations n, number of trees in each random forest l_{tree}, threshold $\epsilon \in (0, 1)$
Output: $\mathcal{D}_t$ with each instance being assigned with a score

Divide $\mathcal{D}_t$ into positive subset $\mathcal{D}_{tp}$ and negative subset $\mathcal{D}_{tn}$
initiate array $\mathcal{SCORE}[\,]$ of length $|\mathcal{D}_p|$
$i \leftarrow 0$
repeat
  $i \leftarrow i + 1$
  for $m = 1$ to l_{tree} **do**
    Randomly sample a subset $\mathcal{P}_{sub}$ from $\mathcal{D}_{tp}$ and a subset $\mathcal{N}_{sub}$ from $\mathcal{D}_{tn}$, where $|\mathcal{P}_{sub}| \gg |\mathcal{N}_{sub}|$
    Learning a classification tree T_m form $\mathcal{N}_{sub} \cup \mathcal{P}_{sub}$
  end for
  for each instance $X_n \in \mathcal{D}_p$ **do**
    for each classification tree T_m in the random forest **do**
      if T_m classify X_n as positive **then**
        $\mathcal{SCORE}[n] \leftarrow \mathcal{SCORE}[n] + 1$
      end if
    end for
    if less than $\epsilon \times l_{tree}$ tree classifiers classify X_n as positive **then**
      Eliminate X_n from future iterations
    end if
  end for
until i=n
Return $\mathcal{SCORE}$ as the output

it is positive. Any instance that receives a score lower than a given threshold will be excluded from the next iteration. At last, all the instances are ranked according to the sum of their scores received in all the iterations.

The idea behind this method is that again we exploit the bias of the classifiers trained from imbalanced data set as in the data cleaning process. Since the positive instances dominate the training data set, each tree is biased towards correctly classifying the positive instance. Then there is a strong possibility for a positive instance to receive a higher score and a negative instance to receive a lower one. By excluding the instances that received a score lower than a specified threshold, we restrict the model to focus on the hard to classify instances.

4 Experimental Evaluation

In this section , we run a series of experiments to evaluate the classification performance of the proposed methods. The decision tree in the random forest is implemented using J48 in the Weka machine learning tool[1].

4.1 Data Set

8 UCI data sets which have different degrees of imbalance were used in our data set. Information of these data are listed in Table 1, including data size, number of attributes, the target attributes, and class distributions. For data sets with more than 1 classes, we choose the class with least instances as the minority class and consider the remainder as the majority class. More details about these data sets can be found in the UCI data page[2].

4.2 AUC Score

The performance of each method is measured by means of ROC [9], which represents the false positive rate on the horizontal axis of a graph and the true positive

Table 1. Information about the data set

Data Set	Size	Attributes	Class	Class Distribution	ClassNumber
balance	625	4	Balance	1:12.0	3
flags	194	28	White	1:10.4	7
haberman	306	3	Die	1:2.81	2
letter	20000	16	A	1:24.3	26
nursery	12960	8	not_recom	1:38.2	5
pima	768	80	1	1:2.01	2
sat	6435	36	6	1:3.23	3
vehicle	846	18	opel	1:3.31	4

[1] http://www.cs.waikato.ac.nz/ml/weka/
[2] http://www.ics.uci.edu/mlearn /MLRepository.html

rate on the vertical axis. A curve is produced by varying the threshold on a classification model's numeric output. The Area Under the Curve is a widely used performance measurement of the classification accuracy on imbalanced data set.

4.3 Comparison with Existing Methods

The proposed methods are compared with 5 other popular techniques used for imbalanced learning, including SMOTE, under-sampling(Under), over-sampling(Over), Tomek links(Tomek), Condensed Nearest Neighbor Rule(CNN). RF refers to our proposed random forest based methods. For each data set, 10 times 10-fold cross validation are executed. In order to reduce the bias introduced by the sampling process, within each fold the learning algorithm is repeated for 20 times. Finally, the averaged AUC score is reported in Figure 2.

RF achieves the best performance on 5 of the data sets. For the other 3 data sets, it is almost as good as the method with the highest AUC score. The similarity function based methods, such as SMOTE and Tomek, does not outperform the simple over-sampling and under-sampling approach since their performance depends greatly on the similarity function and it is not an easy task to find a suitable similarity function for a given data set. While our proposed method does not rely on the similarity function, it outperforms its competitors in terms of prediction performance.

4.4 Influence of Class Distribution

In the classification step, we are reversing the class distribution of the training data for each tree in the forest. In this subsection, we test the influence of the

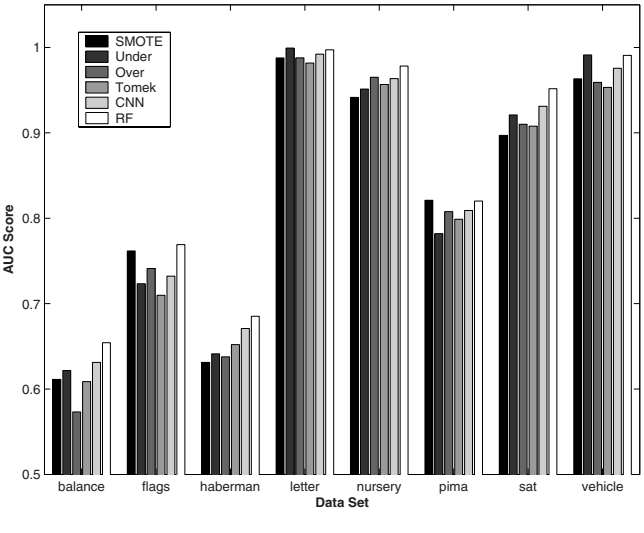


Fig. 2. AUC Scores on 8 UCI Data Sets

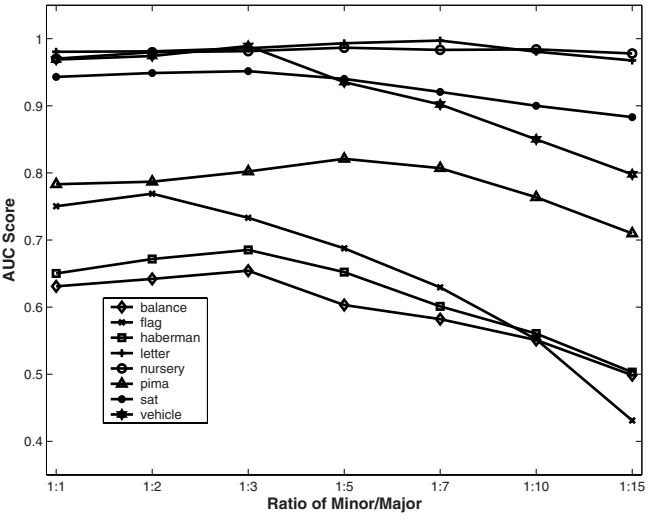


Fig. 3. Influence of Class Distribution

class distribution on the classification accuracy of the proposed method(RF). For each data set, we changed the distribution of the positive instances and the negative instances and tested the performance of the trained model. From Figure 3 we can see that the AUC scores can be largely influenced by the distribution. Our method can achieve the best performance under a certain proper class distribution. As the ration of $Minor/Major$ decreases, the AUC score also decreases.

5 Summary

In this paper , we propose a strategy of learning from imbalanced data set. The main idea of our method is to make the class bias useful when deal with imbalanced data set. In the data cleaning process, the class bias is used to eliminate dangerous negative instances and further address the class overlapping. In the classification process, each classifier is trained to be biased towards the positive class, that is, the class distribution has been reversed. The testing set is classified in an iterative way, where the order in which the instances being labeled reflects their possibility of being positive.

References

1. Breiman, L.: Random Forest. Machine Learning 45, 5–32 (2001)
2. Chawla, N.V., Japkowicz, N., Kolcz, A.: Editorial: Special Issue on Learning from Imbalanced Data Sets. Sigkdd Explorations 6(1), 1 (2004)

3. Ting, K.M.: A comparative study of cost-sensitive boosting algorithms. In: Proceedings of Seventeenth International Conference on Machine Learning, Stanford, CA, pp. 983–990 (2000)
4. Juszczak, P., Duin, R.P.W.: Uncertainty sampling methods for one-class classifiers. In: Proceedings of the ICML 2003 Workshop on Learning from Imbalanced Data Sets (2003)
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 321–357 (2002)
6. Hart, P.E.: The Condensed Nearest Neighbor Rule. IEEE Transactions on Information Theory IT-14, 515–516 (1968)
7. Tomek, I.: Two Modifications of CNN. IEEE Transactions on Systems Man and Communications SMC-6, 769–772 (1976)
8. Aggarwal, C.C.: Towards Systematic Design of Distance Functions for Data Mining Applications. In: Proceedings of the International Conference on Knowledge Discovery and Data Mining (2003)
9. Bradley, A.P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 30(7), 1145–1159 (1997)
10. Bradley, A.P.: Cost-sensitive Learning by Cost-proportionate Example Weighting. In: Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, FL, pp. 435–442 (2003)

Detecting Phishing E-mails by Heterogeneous Classification

M. Dolores del Castillo, Angel Iglesias, and J. Ignacio Serrano

Instituto de Automática Industrial. CSIC, Ctra. Campo Real km. 0.200,
28500 Arganda del Rey. Madrid, Spain
{lola, iglesias, nachosm}@iai.csic.es

Abstract. This paper presents a system for classifying e-mails into two categories, legitimate and fraudulent. This classifier system is based on the serial application of three filters: a Bayesian filter that classifies the textual content of e-mails, a rule- based filter that classifies the non grammatical content of e-mails and, finally, a filter based on an emulator of fictitious accesses which classifies the responses from websites referenced by links contained in e-mails. This system is based on an approach that is hybrid, because it uses different classification methods, and also integrated, because it takes into account all kind of data and information contained in e-mails. This approach aims to provide an effective and efficient classification. The system first applies fast and reliable classification methods, and only when the resulting classification decision is imprecise does the system apply more complex analysis and classification methods.

Keywords: e-mail classification, web filtering, multistrategy learning.

1 Introduction

Phishing is the term used to describe massive e-mails that trick recipients into revealing their personal or company confidential information, such as social security and financial account numbers, account passwords and other identity or security information. These e-mails request the user's personal information as a client of a legitimate entity with a link to a website that looks like the legitimate entity's website or with a form contained in the body of the e-mail. The aim of phishing is to steal a user's identity in order to make fraudulent transactions as if the phisher were the user.

According to Anti-Phishing Working Group [9] the number of phishing reports increased from 20,109 in May 2006 to 28,571 in June 2006 to make it the most ever recorded. Phishing attacks are increasing despite of the use of e-mail filters. Although only 0.001 percent of these e-mails sent are answered, this percentage is enough to provide a return on investment and keep the phishing industry alive [26].

The next section reviews different filter types currently available. Section 3 gives details of the anti-phishing system proposed. Section 4 discusses the empirical evaluation of this approach. Section 5 focuses on the learning ability of the system, and the final section presents the conclusions and point the way to future work on this subject.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 296–305, 2007.

2 Phishing Filtering

Information on techniques for avoiding electronic fraud is not readily available in the scientific and marketing literature. This lack of information prevents phishers from hitting upon for the circumvention of filters.

Nowadays, although different systems exist to deal with the problem of electronic fraud, these systems are far from optimal for classification purposes. The methods underlying these systems can be classified into two categories, depending on the part of the message chosen by the filter as the focus for deciding whether e-mail messages are legitimate or fraudulent:

1. Origin-based filtering. Origin-based filters focus on the source of the e-mail and verify whether this source is on a white verification list [8] or on a black verification list [1], [29].
2. Content-based filters focus on the subject and body of the e-mail. These can be also divided into two classes, depending on whether the analyzed content is textual or non-textual. Textual content filters classify e-mails using Bayesian classifiers [2], [7], [10], heuristic rules [23], [28] or a combination of them [10]. Non-textual content filters examine the links embedded in e-mails by different techniques. Some filters check whether the links belong to white and black verification lists [6], [10]. Other filters analyze the appearance of the alphanumeric string of links and look for well known phishing schemes [14], [15]. In [4], a combination of internal and external information relative to both textual content and links in e-mails is used to build an effective e-mail filter.

Besides the phishing e-mail filters, there exist other anti-phishing tools, the toolbars, which can be built into browsers for blocking access to or warning the user about web pages identified as possible or actual phish. Examples of these web filters are [5], [16], [25]. In [18] a comparative test of several toolbars is shown.

Most current commercial e-mail filters use verification lists to analyze e-mail senders and links contained in the body of e-mails. These filters are dynamic and update verification lists when new attacks are reported. However, the updating rate of filters is often overcome by the changing rate of the attacks because phishing e-mails are continuously modifying senders and link strings, many websites are only available one day, and continuous updating thus implies a high economic cost.

This paper describes a client-side system, which was designed and built to detect and filter phishing e-mail automatically, using different sources of information present in the content of e-mails which are handled by the processing methods most suitable for each information type.

3 Classification System

The range of procedures used by a phisher, which evolve quickly in order to evade filters available, makes it necessary to solve the problem of identifying phishing e-mails with a multistrategy and integrated approach. The system shown in this paper focuses on a global view of all the information provided by an e-mail and the different analysis methods for each kind of information.

One main feature of this system is effectiveness, measured by *precision* (percentage of predicted documents for a category that are correctly classified) and *recall* (percentage of documents for a category that are correctly classified), because it applies three different classifiers in order to obtain all the information needed at each decision point to classify an e-mail serially. Another relevant feature is efficiency, because the order in which the classifiers are applied aims to obtain fast and reliable classifications by minimizing the resources used in a first classification step and then, if it is necessary, more complex methods are used in following classification steps.

The system decides to assign an e-mail to the *Fraud* or *Legitimate* categories after applying first, a Naïve Bayes classifier that focuses on the textual content of e-mails and assigns them to the *Economic* or *Non-Economic* categories. It should be highlighted that it is very difficult to obtain financial legitimate e-mails because neither the financial sector nor users give their legitimate economic e-mails to generate legitimate economic corpuses that can be analyzed by the scientific community. Besides, the non-on-line financial entities are reluctant to use such means primarily due to the growing number of phishing attacks. This lack makes it difficult to train Bayesian classifiers that discriminate between the *Legitimate* and *Fraud* categories according to the textual content of e-mails. This is the reason for training both Bayesian classifiers on a corpus of e-mails labeled into the *Economic* and *Non-Economic* categories.

Next, a rule-based classifier, which focuses on non grammatical features of e-mails, classifies e-mails previously assigned to the *Economic* category by the first classifier into one of the *Legitimate*, *Fraud* or *Suspicious* categories. Last, a third classifier emulates a fictitious access to websites referenced by links contained in the body of the e-mails assigned to the *Suspicious* category by the rule-based classifier. This classifier analyzes the responses obtained from the fictitious access to these websites and classifies them into the *Fraud* or *Legitimate* categories.

The system includes a bias against generating false negatives, i.e., e-mails erroneously classified in the *Legitimate* category, and false positives, i.e., e-mails erroneously classified in the *Fraud* category. When a classifier lacks information at some decision point, the classifier assigns the e-mail to the category that allows the system to further analyze the e-mail at a deeper level. The bias tries to enforce the user's safety against false negatives and classification performance of the system against false positives. Although the outcome from wrongly classifying an e-mail as legitimate is more dangerous for the user than classifying an actual legitimate e-mail as fraudulent, the system takes into account both kinds of misclassifications.

The system can learn incrementally from past mistakes when the user prompts misclassified e-mails by an interface that shows the final classification results. The architecture of this classifier system is modular and flexible. This easily allows new analysis and detection methods to be added as they are developed.

3.1 Naïve Bayes Classifier for the Textual Content of E-mails

This classifier was developed to identify and filter e-mail based on the Naïve Bayes statistical classification model [12], [21]. Since there are two parts in an e-mail, subject and body, which can contain text, a Bayesian classifier is built for each part. In the training stage, the probabilities for each word conditioned to each category are

estimated, and a vocabulary of words with their associated probabilities is created. The filter classifies a new text into a category by estimating the probability of the text for each possible category C_j, defined as $P\ (C_j \mid text) = P\ (C_j)\ .\ \Pi_i\ P\ (word_i \mid C_j)$, where $word_i$ represents each word contained in the text to be classified. Once these computations have been performed, the Bayesian classifier assigns the text to the category that has the highest probability value.

In the classification phase of a new e-mail, the Bayesian classifier composes a word vector associated with each part of the e-mail and computes the economic probability *Pec* and non-economic probability *Pnec* for each part. The parts are assigned to the category with the highest probability. The system integrates the classification decisions of both parts by assigning the e-mail to the *Economic* category whenever the classification result of any of the two parts is in the *Economic* category. This course of action claims to avoid generating false negatives.

3.2 Rule-Based Classifier for the Non Grammatical Content of E-mails

Since knowing whether an e-mail is economic is not sufficient to classify it into the *Legitimate* and *Fraud* categories, the goal of the system is only attained by analyzing other kinds of data contained in e-mails previously classified into the *Economic* category by the Bayesian classifier. This analysis aims to find out the values of the discriminatory features present in economic e-mails: links, images, and forms. An algorithm based on AQ learning [11] trained on a corpus of fraudulent and legitimate economic e-mails, which are represented by these three features, builds the descriptions and thus the rules for each category.

The first rule states that if the body of an economic e-mail does not contain forms, images or links, then the e-mail is assigned to the *Legitimate* category. The second rule determines that an e-mail that requests information directly from a form contained in the body is not safe at all. Accordingly, the classifier categorizes all e-mails dealing with economic topics and not offering any security when requesting personal information into the *Fraud* category. Since both fraudulent and legitimate e-mails can contain images or links, these features do not determine with certainty the category of e-mails. In order to avoid generating false positives, the rule-based classifier uses a third rule to decide whether an e-mail is potentially dangerous to users. This rule states that if an e-mail deals with economic topics and it contains links in the text or links in an image, then the danger could come from the websites referenced by these links. So, the classifier classify the e-mail into the *Suspicious* category and a third classifier further processes the e-mail. An e-mail categorized as *Suspicious* is considered by the system lacking of knowledge to make a decision. Each rule has a dynamic confidence level. This confidence level can be modified as the system learns from misclassifications.

3.3 A Classifier Based on an Emulator for the Content of Websites Addressed by the Links Contained in E-mails

This classifier processes the links contained in the body of the e-mail and assigns a suspicious e-mail to the *Legitimate* or *Fraud* categories in a procedure consisting of three steps: 1) a meta-searcher extracts the structure of the websites addressed by

links, 2) an emulator fills in the website structure with fictitious data, and 3) a finite state automaton tries to recognize the answer given by the website.

The meta-searcher is a procedure for obtaining the links contained in e-mails and extracting the patterns of the forms contained in the websites referenced by these links. First of all, the meta-searcher verifies whether the websites actually exist using an Internet search engine. Since fraudulent websites are available for a short period of time, this classifier determines that if there is no information about a website, then there is a high probability that the website is fraudulent. So, the e-mail that references this website is classified into the *Fraud* category.

If the website exists, then the meta-searcher verifies that the accessed website uses a secure *http* connection (*https*). If a website is not safe, then the e-mail is classified into the *Fraud* category. If the website exists and it is safe, then the meta-searcher looks for the forms contained in the website and extracts all the fields that need to be filled in. The meta-searcher analyzes the kind of data of every field and detects whether a website requests sensitive information, like a password. If a website does not request personal information, then the meta-searcher determines that this website is legitimate and thus, the e-mail is classified into the *Legitimate* category.

If a website requests personal information, then the emulator fills in the forms using fictitious data and then it submits the information in order to obtain the website response. At this point, the website response needs to be analyzed to know its nature and classify the e-mail.

The classification model used for recognizing the response given by the website is based on the construction of a finite state automaton. The automaton represents words contained in a certain language and collects the grammar presented in this language. The language used in this case consists of the sentences contained in the responses given from legitimate financial websites to fictitious accesses. All of these responses warn the user of an error that occurred while processing the information requested by the website with a different vocabulary of words and a different grammar. The automaton is a generalized description of the instances of the class "legitimate responses to fictitious accesses". Fictitious access to fraudulent websites return no error message since their sole goal is to collect users' confidential information. This classification model allows a high degree of generalization in the classification and it is very efficient because it is not time-consuming and uses few resources.

A grammar inference algorithm called ECGI [19] takes the sentences and the probabilities of all bigrams (groups of two words) and generates the automaton states and the edges between the states. Every edge connecting two states is weighted with the probability of the corresponding bigram. When the automaton is going to recognize a new sentence, the words of the sentence are processed according to their written order [22]. If there is a state representing the processed word and this state is reachable from the actual state, then the automaton follows that connection. If there is no state reachable representing the word, then the automaton follows the connection that has the highest probability. After following this connection, the automaton processes the next word. When the automaton reaches the final state, the automaton response is the sequence of words with the highest probability and similarity related to the sequence of words included in the sentence analyzed.

The emulator-based classifier uses a function that computes the similarity between an input sentence and the sentence obtained by the automaton by calculating the

relation between the number of overlapping words in both sentences, and the total number of words in each sentence. If the minimum of the two percentages calculated is greater than or equal to 50%, then it is considered that the two sentences are similar. If the minimum is less than 50%, then the function computes the sum of the document frequency (the number of sentences contained in the base of responses which include that word) of the overlapping words in order to give a higher score to the words that appear in more sentences of the base of responses. If the sum of the document frequency of the overlapping words is greater than or equal to a threshold, whose value has been empirically determined, then the classifier considers that the two sentences are similar. If the sum is less than the threshold, then the sentences are not considered similar.

Therefore, if the emulator-based classifier considers that a textual response obtained after a fictitious access is similar enough to the automaton response, then the textual response is considered legitimate. So the e-mail that references the website is classified into the *Legitimate* category. If the responses are not similar enough then the e-mail is classified into the *Fraud* category.

4 Empirical Evaluation

The classifier system was evaluated on a set of messages. This dataset is composed of 1,038 economic messages, divided into 10 legitimate messages and 1,028 fraudulent messages, and 1,006 non-economic messages. A small fraction (4 e-mails) of the legitimate messages, i.e., economic messages coming from legitimate financial entities, and all non-economic messages were extracted from [24]. The remaining legitimate messages (6) were received by the authors of this paper during a given period of time. From 1,028 fraudulent economic messages, 833 were extracted from [13] and the remaining messages came from the inboxes of the authors of this paper. In the evaluation shown in this paper, messages from a sub corpus of [24] were carefully analyzed by hand for extracting the non-economic messages. Currently, the vocabulary of terms needed by the Bayesian filter for classifying messages into the economic or non-economic categories has been learned from the textual contents of two corpus of webpages dealing with economic topics and a third corpus dealing with general topics.

The Bayesian classifiers were trained and tested on this dataset using 5-fold cross validation, that is, four folds were the training data and the fifth was the test data. The content of both textual parts of messages was preprocessed by applying a stop list, and stemming the words [17]. Next, words recurring below an experimentally determined threshold value of the function tf.idf [20] were removed. All words were sorted by the Chi-square statistical measurement [20] and only the 50% highest ranked words of the vocabulary of each part were retained. After that, both textual Bayesian classifiers were built.

Table 1 shows the *precision* (*Pr*), *recall* (*Rc*) and *F*-measure (*F*) (F = (2 * precision * recall) / (precision + recall)) values obtained by the integration of both Bayesian classifiers for the *Economic* (E) and *Non-Economic* (N-E) categories. The last column presents the macro averaged values and the other five columns (Si) present the result of each execution of the cross validation.

Table 1. Performance measurements resulting from integrating the classifications obtained by the two textual Bayesian classifiers (Categories: *Non-Economic* N-E*, Economic,* E)

	S1	S2	S3	S4	S5	Average
Pr N-E	0.956	0.976	0.995	0.995	0.926	0.969
Rc N-E	0.975	0.995	0.985	0.975	0.683	0.922
F N-E	0.965	0.985	0.989	0.984	0.786	0.942
Pr E	0.975	0.995	0.986	0.976	0.753	0.937
Rc E	0.957	0.976	0.995	0.995	0.937	0.972
F E	0.965	0.985	0.990	0.985	0.834	0.952

A supervised learning algorithm trained on the fraudulent and legitimate e-mails of the dataset generated the decision rules for the *Fraud*, *Legitimate* and *Suspicious* categories. These rules were used to classify all the e-mails in the five folds which were previously categorized as *Economic* by the Bayesian classifier. In a first experimental setting, suspicious e-mails were classified in the *Fraud* class by default. The effectiveness values of this classifier for the *Fraud* (F) and *Legitimate* (L) categories is reported in Table 2.

Table 2. Performance measurements of the serial application of the Bayesian and rule-based classifiers (Categories: *Fraud* F, *Legitimate* L)

Pr F	*Rc* F	*F* F	*Pr* L	*Rc* L	*F* L
0.941	0.949	0.944	0.943	0.921	0.930

In order for the system to make a confident decision about the class of a message, messages assigned to the *Suspicious* class were finally processed by the emulator-based classifier.

A base of responses was built to create the finite state automaton used by the emulator classifier. This base collects responses from different legitimate financial websites to fictitious accesses. These responses were preprocessed by using a stop list and a stemming algorithm [17]. The stemming algorithm does not reduce verbal endings that are used to determine verb tenses ("*Introduce* the information" is different from "The information *introduced*").

In the classification phase of the test responses, the classifier did not take the whole sentences obtained from accessing. Instead, it considered groups of three consecutive words. The choice of using groups of three words is based on the fact that most of the sentences included in the base of responses consisted of three words.

Table 3. Performance measurements of the serial application of the Bayesian, rule-based and emulator-based classifiers (Categories: *Fraud* F, *Legitimate* L)

Pr F	*Rc* F	*F* F	*Pr* L	*Rc* L	*F* L
0.962	0.948	0.955	0.949	0.962	0.955

The third classifier classifies the messages previously assigned to the *Suspicious* class by the rule-based classifier. Table 3 shows the overall performance measurements of the system for the *Fraud* (F) and *Legitimate* (L) categories. All effectiveness values are better than those obtained by only applying the Bayesian and the rule-based classifiers. The measurements indicate that this third classification step produces improvements of the precision value for the *Fraud* class and the recall value for the *Legitimate* class because it allows the system to classify more certainly the e-mails containing links and thus to generate less false positives.

5 Evolution of the System

Fraudulent e-mails will continue to evolve in an attempt to evade the filters available. In order to prevent increased misclassification and, consequently, reduced classification performance, the system has to be able to learn from past mistakes. Classification errors can be of two types: 1) false positives or *False Fraud*, which are e-mails incorrectly classified as fraudulent when they are legitimate, and 2) false negatives or *False Legitimate*, which are fraudulent e-mails incorrectly classified as legitimate.

As highlighted in previous sections, the system applies a bias that tries to protect users from the most dangerous error, the *False Legitimate*. Users are putting themselves at risk if they decide to access to an email classified into *Legitimate* class when it is actually *Fraud*. This is the reason why the system learning ability focuses on *False Legitimate* e-mails. When users identify a *False Legitimate*, they can prompt the e-mail misclassified by an interface designed for this purpose. The decision about this wrong classification can be caused by any of the three classifiers in the overall system. In order to know which classifiers are responsible for the misclassification, the system must save a classification track of the decisions taken by all the classifiers. Thus, the classifier that made the error can be identified. Depending on the classifier that wrongly decided, the learning ability differs:

1) If the Bayesian classifiers assign an economic e-mail into the *Non-Economic* class, both vocabularies are updated in terms of the number of words and word probabilities associated with the economic and non-economic categories.

2) The rule-based classifier assigns a fraudulent e-mail into the *Legitimate* class when there are no signs of danger present in this e-mail, i.e., the e-mail does not contain any form or link. In this instance, the classifier decreases the confidence level of the rules implied in the classification by an empirically determined amount.

3) The emulator-based classifier assigns a suspicious e-mail to the *Legitimate* class when it is actually a fraudulent e-mail, because the automaton has not recognized correctly the responses to fictitious accesses to the websites referenced by the links contained in the suspicious e-mail. Here, learning implies that the sentences wrongly recognized by the automaton are added to a sentence stoplist. When the automaton receives a sentence, it verifies whether such sentence is included in the stoplist. If it is, the automaton does not process it and further analyzes the remaining sentences in the response.

6 Conclusions and Future Work

The system presented in this paper is based on a hybrid approach that takes advantage of applying different processing methods to multiple data sources. It is an effective system for avoiding the creation of false positives and negatives. The major novelty comes from the emulator method used to classify the webpages addressed by the links contained in e-mails. The analysis of all kind of data present in e-mails allows the classification to be independent of external information sources, like verification lists or web reputation servers, leading to a more efficient decision making.

Besides the learning capabilities of the system, thanks to its modular and flexible design, it does allow for straightforward upgrading as new processing methods become available or new features included in future phishing attacks can be identified.

Currently, this approach is being easily extended to build webpage classifiers. Here, the webpage textual content is categorized as economic or non-economic by integrating the predictions made by several classifiers that categorize each one of the four possible textual parts (url, meta-text, plain-text, links) of a webpage [3]. If the webpage is economic and it contains forms requiring confidential information, the emulator fills in the forms with fictitious data and it submits them. Next, the answer obtained is analyzed and classified and accordingly the webpage is also categorized.

References

1. Aladdin eSafe (2005), http://www.aladdin.com
2. Androutsopoulos, I., Paliouras, G., Karkaletsis, G., Sakkis, G., Spyropoulos, C., Stamatopoulos, P.: Learning to Filter Spam E-mail: A Comparison of a Naive Bayesian and a Memory-Based Approach. In: Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (2000)
3. Castillo, M.D., Serrano, J.I.: A Multistrategy Approach for Digital Text Categorization from Imbalanced Documents. ACM SIGKDD Explorations 6, 70–79 (2004)
4. Fette, I., Sadeh, N., Tomasic, A.: Learning to Detect Phishing Emails. In: WWW 2007, Banff, Canada (2007)
5. GeoTrust TrustWatch (2004), http://www.trustwatch.com/
6. GoDaddy (2006), http://www.godaddy.com/
7. Graham, P.: Better Bayesian Filtering. In: Proc. of Spam Conference 2003, MIT Media Lab., Cambridge (2003)
8. Iconix eMail ID (2005), http://www.iconix.com
9. June Phishing Activity Trends Report (2006), http://www.antiphishing.org
10. McAffee SpamKiller (2003), http://www.spamkiller.com
11. Michalsky, R.S.: A Theory and Methodology of Inductive Learning. In: Machine Learning: An Artificial Intelligence Approach, pp. 83–134. Springer, Heidelberg (1983)
12. Mitchell, T.M.: Machine Learning. McGraw-Hill, New York (1997)
13. Monkey.Org Inc. (2006), http://www.monkey.org/~jose/wiki/doku.php
14. Microsoft Outlook 2003. SP 2. (2005), http://office.microsoft.com
15. Mozilla Thunderbird 2 (2005), http://www.mozilla.com/thunderbird
16. Netcraft (2007), http://news.netcraft.com/

17. Porter, M.F.: An Algorithm for Suffix Stripping. Program 14(3), 130–137 (1980)
18. Robichaux, P., Ganger, D.L.: Gone Phishing: Evaluation Anti-Phishing Tools for Windows, 3Sharp LLC (2006)
19. Rulot, H.: ECGI. Un Algoritmo de Inferencia Gramatical mediante Corrección de Errores. Phd Thesis, Facultad de Ciencias Físicas, Universidad de Valencia (1992)
20. Salton, G., Buckley, C.: Term-weighting Approaches in Automatic Text Retrieval. Inform. Proccesing & Management 24(5), 513–523 (1988)
21. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys 34(1), 1–47 (2002)
22. Serrano, J.I., Araujo, L.: Statistical Recognition of Noun Phrases in Unrestricted Texts. In: Famili, A.F., Kok, J.N., Peña, J.M., Siebes, A., Feelders, A. (eds.) IDA 2005. LNCS, vol. 3646, pp. 397–408. Springer, Heidelberg (2005)
23. Sophos Email Security and Control (2005), http://www.sophos.com
24. SpamAssassin (2006), http://spamassassin.apache.org/publiccorpus/
25. Spoofstick (2005), http://www.spoofstick.com
26. Suckers for spam (2005), http://www.Internetnews.com
27. Tagged Message Delivery Agent Homepage (2006), http://tmda.net
28. Tumbleweed MailGate Email Firewall (2006), http://www.tumbleweed.com/
29. Verisign Messaging security (2006), http://www.verisign.com

Load Balancing in Fault Tolerant Video Server

D.N. Sujatha[1], K. Girish[1], B. Rashmi[1], K.R. Venugopal[1], and L.M. Patnaik[2]

[1] Department of Computer Science and Engineering
University Visvesvaraya College of Engineering Bangalore University
Bangalore-560001, India
suj_sat@yahoo.com
[2] Microprocessor Applications Laboratory, Indian Institute of Science
Bangalore-560012, India

Abstract. The Video-on-Demand (VoD) application is popular as the videos are delivered to the users at anytime and anywhere. The system load is increased due to simultaneous access of VoD system by many users. The proposed architecture discusses load balancing mechanism within a video server and to provide service to the users with small start-up delay. The Video storage is based on the probability of the users requesting for the video. Videos with higher request probability are stored and replicated to ensure guaranteed retrieval. Parity generation scheme is employed to provide reliability to non-popular videos. The system is also capable of handling disk failures transparently and thereby providing a reliable service to the user.

Keywords: Data Striping, Fault Tolerance, Load Balancing, Video Server.

1 Introduction

Video-on-Demand (VoD) is a real time multimedia application in which videos are stored in the video server, assisting in delivery of video requested at any time under the discretion of the user. Implementation of VoD is not as widespread as Internet with its biggest hindrance being the lack of network infrastructure that can handle large amount of multimedia data. There is need for design mechanisms that can determine how best to exposit upon existing infrastructure in order to support the additional demand inflicted by VoD systems. These mechanisms should strive towards generating scalable architecture. Load Balancing technology addresses these issues making the system scalable and better performing. *Load balancing* is a technique to spread work between many disks in order to get optimal resource utilization and decrease computing time thereby providing faster service to the users. The challenge for researchers is to design mechanisms to support more users and to reduce the average response time. The load balancing among the servers can be achieved if a load balancing mechanism is triggered to perform file migration and request migration when load unbalance is detected. To perform these activities an external hardware device, load balancer is used. A load balancer

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 306–315, 2007.

is capable of increasing the capacity of a server farm beyond that of a single server. It also allows the service to continue even during server down time due to failure or maintenance. However there are limitations to this approach. Load balancers can become points of failures and performance bottlenecks themselves. Hence, load balancing is performed within the video server rather than maintaining external devices. As the number of servers grows, the risk of a failure at any point in the system increases and must be addressed. The load balancers are capable of balancing the load as well as detecting the failures. Multiple disks in video server significantly improve storage capacity and I/O bandwidth. However, the use of multiple devices increases the probability of failure. In a VoD system, without fault tolerance, a failure would result in a disruption of service to all users. A VoD system should provide a high degree of fault-tolerance. If a disk in the server fails, load balancing automatically redistributes requests to other disks within the disk array or forward the request to other servers in the cluster, hence ensuring continuous service to the users. Server load balancing addresses several issues such as: increased scalability, high performance, high availability and disaster recovery.

Outline: The reminder of this paper is organized as follows. Section 2 presents previous works in the area of load balancing in VoD system. Load balancing Architecture for VoD System is discussed in Section 3. The Algorithm is discussed in Section 4. We elucidate the usefulness of our algorithm through simulation and performance analysis in Section 5 and conclusions in Section 6.

2 Previous Works

Load balancing can be performed using external devices in VoD servers. Niyato et al. [1] describes external load balancing performed in Internet video and audio server. Load balancing in a video server can be carried out with the help of disk arrays in the server. Various techniques are employed to efficiently distribute the data among these disks to ensure that the users are served faster with minimum access times. One of such techniques employed is RAID (Random Array of Inexpensive Disks). There are many levels of RAID existing to store data in reliable fashion. RAID-0 (disk striping) and RAID-1 (disk mirroring) and RAID-5 (disk striping with parity) are few of the popular RAID levels. Lee et al. [2] employs the disk striping with parity in video servers to provide load balancing along with fault tolerance. This scheme suffers with multi-disk failures. A comparison of different RAID levels evolved Random Duplicate Assignment (RDA) [3]. In this strategy the video is striped and instead of being stored sequentially they are randomly allocated in different disks and each strip is mirrored to enable fault tolerance. This scheme enables the video server to balance the load among its disks and also achieve reliability. The time required to access next block in the disk increase as the blocks are randomly allocated. A map between the blocks is to be maintained to handle this problem, which is an additional overhead. Replicating all the videos is considered in this scheme, which may not be feasible all the time. Golubchik et al. [4] discusses fundamental issues associated

with providing fault tolerance in multi-disk VoD servers. Storage of videos in a video server may be characterized by popularity of the video. Replication of videos and placement of video blocks based on popularity is discussed in [5], [6] and explores a data placement method based on rate staggering to store scalable video data in a disk-array based video server.

3 System Architecture

The storage scheme designed, evaluates the popularity of the video before it is stored. The video storage differs with popularity. If the video is popular, then it is striped across array of disks and mirrored else it is striped across the disks with the last disk in the set of disks to store the parity information. This helps to rebuild the video block in case of disk failure. The overview of the architecture is shown in Fig. 1.

- *Request Dispatcher* is a controller that is required at the user side to take the user's command and send the signal to the server through its network interface. The request dispatcher also stores video signals it receives from the server into its buffer, decodes the compressed signals and sends the decoded signals to the display at the appropriate time.
- *Buffer* is the video memory that holds the video image to be displayed on the screen of the user. The amount of memory required to hold the image depends primarily on the resolution of the image and also the color depth used per pixel. Once the server locates the video and moves the video frame by frame to the user, the frames are first stored temporarily in the user's buffer before it is displayed to the user. The advantage of having a buffer in the user's memory is to reduce the jitter and to provide high quality video transmission to the users.

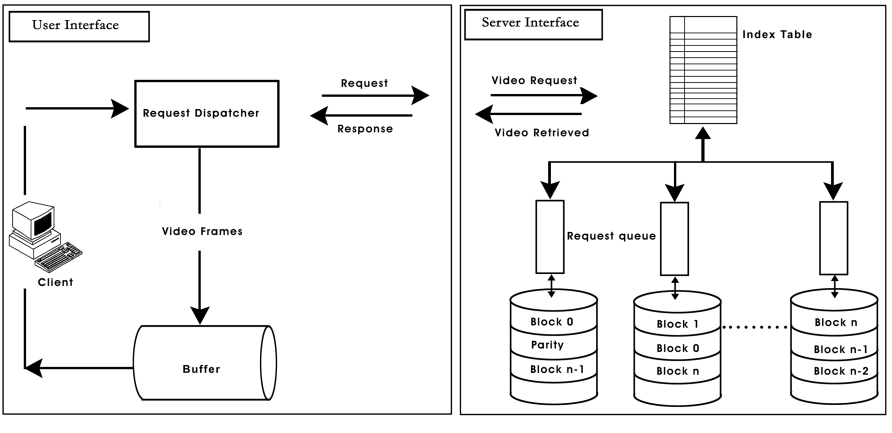


Fig. 1. Overview of the Architecture

- *Index Table* maintains the list of unique id of all the videos stored in the current video server. When a new video is stored, index table is updated. When a request arrives for the video, index table is searched to find if the requested video is available and if available it retrieves the video from disk and begins streaming.
- *Disk Array* is used to store videos. Disk array refers to a linked group of one or more physical independent hard disk drives generally used to replace larger, single disk drive systems. Disk arrays incorporate controls and a structure that pre-empts disaster. The most common disk array technology is RAID. RAID utilizes disk arrays in a number of optional configurations that benefit the user. Disk arrays organize their data storage into Logical Units (LUs), which appear as linear block places to their users. A small disk array, with a few disks, might support up to 8 LUs; a large one, with hundreds of disk drives, can support thousands.

The storage mechanism for popular videos is shown in Fig. 2. The video is divided into blocks based on number of disks. Each block is stored in different disks sequentially so that only once a block of video stored on the disk. The first disk to store the video is rotated to ensure that the load is balanced among the disks. If the block requested is stored in a disk, which is serving another user, the request is queued. The requests in the request queue are serviced in round robin fashion. If the video is a popular then, the video is stripped across the array of the disks and mirrored. In case of increase of load for the popular video the request may be served by the mirrored storage. The mirrored data is also accessed in case of disk failure containing the popular videos. The storage mechanism for non-popular videos is shown in Fig. 3. If the video is non-popular video, it is striped across the disks with the last disk in the set of disks to store the parity information. Performing XOR operation between all the blocks of data generates parity block. This helps to rebuild the video block in case of disk failure.

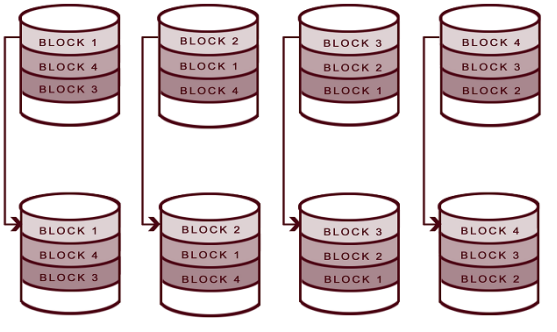


Fig. 2. Storage Mechanism for Popular Videos

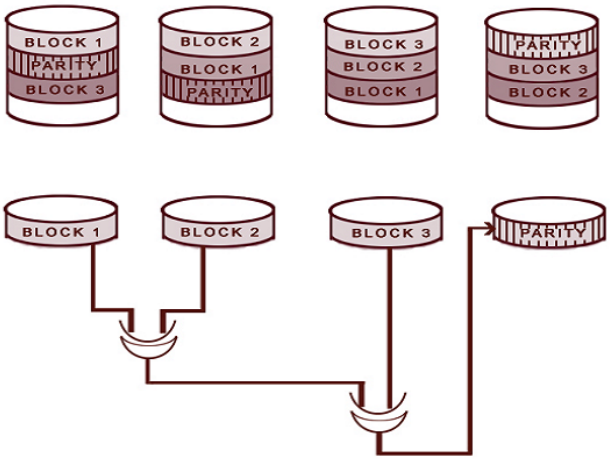


Fig. 3. Storage Mechanism for Non-Popular Videos

Table 1. Video Storage

```
Video_Storage (VideoId)
begin
determine the popularity of the video.
   if (video is popular)
     BlockSize = VideoSize / n
   else
     BlockSize = VideoSize / n-1
   i=0
  j = 0
  if ( space available to store video at BlockSize on disk j)
    begin
      until ( VideoSize≥ 0)
      begin
        j = (( VideoId % n )+ i)% n
        reduce VideoSize by BlockSize.
        store block on diskj
        increment i by 1
      end
    end
  Update_Index_Table (VideoId)
  if (video is not popular)
    P_j = B_0 ⊕ B_1
      hile(i <   )
        _j =  _j ⊕ B_i
    Store P_j in disk j
   else
     mirror the blocks in mirror storage.
end.
```

4 Algorithm

The storage routine (Refer Table 1) shows how the videos are stored in the server to facilitate load balancing and provide fault tolerance feature. The video is checked for popularity using Zipf distribution. If the video is popular the block size is determined by total video size by *N* (number of disks). If the video is non-popular, its block size is determined by video size by number *N-1* (disks). Here, one disk is used to store the parity information. Rotational storage scheme is used to store the video blocks. The first block of the first video is stored sequentially from first disk. The first block of next video is stored from second disk. This enables load distribution when simultaneous requests for various videos are made. The blocks thus stored are either replicated in mirrored storage area for popular videos or parity block is generated for non-popular videos. The index table is then updated to store video information. It holds information regarding video file-id, the size of the video and the disk in which the first block of the video is stored.

The HandleRequest routine (Refer Table 2) is designed to illustrate the behavior of the server on arrival of the request. When the server receives a request for a video with id, the server checks the video-id against the video-id of the available list of videos in the index table. If the requested video-id is found, it retrieves the information indicating to the disk in which the first block of video is present. Before the video frames are retrieved, it is necessary to check whether the block is not corrupted and is retrievable. If the block of popular videos is

Table 2. Video Retrieval

```
Handle_Request (Video_id)
begin
  found = search index table for VideoId
  if (found)
    retrieve video_info from index table.
  until (EOF)
  begin
    if (block not corrupted)
    begin
      if (disk not loaded)
        begin
          stream block i from disk j
          j = j+1
        end
      else if (video is popular)
        handle request from replica disk
      else
        Forward_Request (Video_Id)
      end
  else if (video is not popular)
      rebuild block from parity block
  else
      stream from replica.
  end
end.
```

corrupted, the request is handled by the mirrored disks. If the block of non-popular video is corrupted, the block is retrieved with the help of special parity block available. The block of video is streamed from mirrored storage, if request queue of the disk is full. If no mirrored data is available, then the request is forwarded to other server in server farm.

5 Simulation and Performance Analysis

Fig. 4 illustrates the start-up delay of the users to begin downloading the video after the request is made. The average delay increases with the increase in load,

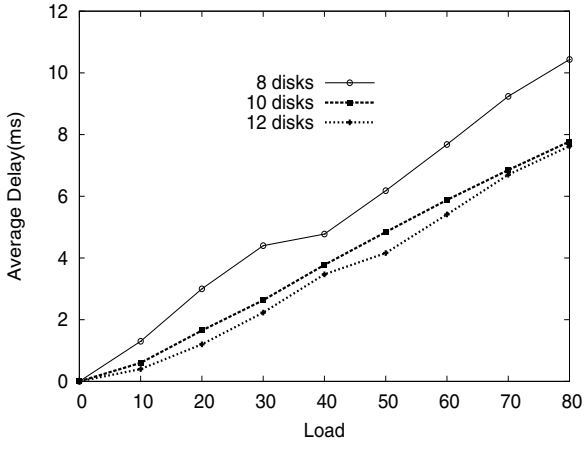


Fig. 4. Start-up delay

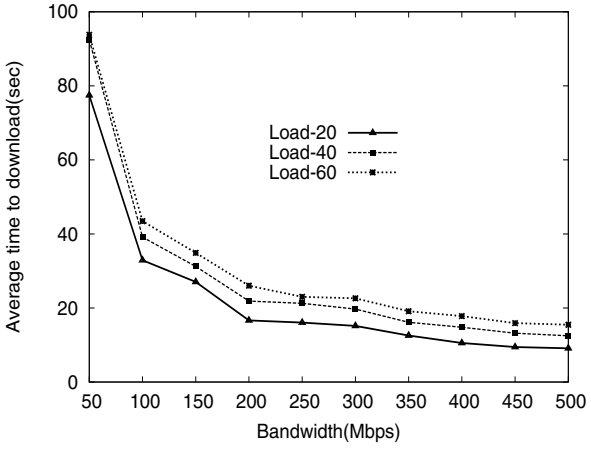


Fig. 5. Bandwidth Utilization for Varying Load

as the users are queued if the disk is busy serving other users. The delay increases by 17% with increase in load. The start up delay can be further decreased with increase in number of disks in the system which is evident from the graph. Fig. 4 also depicts the decrease in start up delay with increase in number of disks.

Bandwidth utilization for varying load is ascertained in Fig. 5. The increase in bandwidth decreases download time of the entire video. Variations of load increase the time to download by 2% to 4%. Increase in bandwidth reduces the time to download by 75% reaching saturation at higher bandwidth rates.

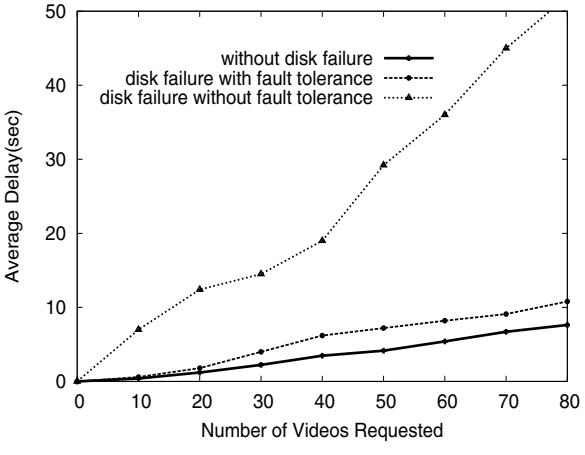


Fig. 6. Performance of System with Disk Failure

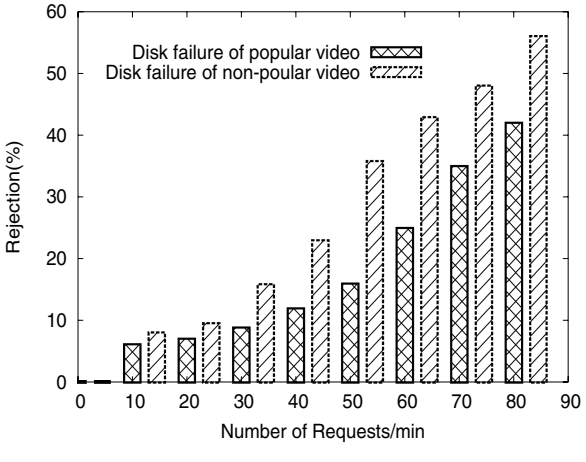


Fig. 7. Illustration of Disk Failure with Popular and Non-popular Videos

The performance of system with disk failure is shown in Fig. 6. Delay is initially the same with disk failure and a normal system. It increases gradually with the increases in load by 2 to 2.5% and the difference remains constant. A drastic increase can be noted if the fault tolerance mechanism was not implemented. Popular videos have a higher tolerance to load than non-popular videos. As the load increases the percentage of rejection increases in non-popular videos by 25%.

Fig. 7 illustrates the disk failure of the popular videos against the non-popular videos. The popular videos demonstrates lower rejections thereby increasing the throughput of the system.

6 Conclusions

An efficient architecture has been proposed to balance the load in the video server. Storage mechanism to distribute the load in an array of disks are considered. The video in divided to blocks and these blocks are stored on different disks in disk array along with disk rotation. The video server is ensured to be fault tolerant by using both mirroring and parity schemes. The selection of fault tolerance scheme is based on popularity of the video using mirroring and thus providing back-up to handle load imbalance and disk failure situations. In case of a non-popular video the probability of failure is less. Nevertheless to provide reliable service, in case of non-popular video single disk failure is handled using parity blocks. The simulation and resultant graphs depict a high system throughput of 100% up to 30 simultaneous requests. There is a gradual increase in the rejection rate beyond 30 requests. The system also has a very low latency of 2ms-10ms for a varying load up to 80 requests. The increase in number of disks further decreases the latency. The system provides fault tolerance against disk failures with a very low performance degradation. Hence the proposed architecture distributes the load in the video server efficiently to handle simultaneous requests and provides high degree of fault tolerance there by enabling efficient and effective service to the users requesting the video.

References

1. Niyato, D., Srinilta, C.: Load Balancing Algorithms for Internet Video and Audio Server. In: 9th IEEE Intl. Conference on Networks, pp. 76–80 (2001)
2. Lee, Y.B., Wong, P.C.: VIOLA-A Scalable and Fault-Tolerant Video-on-Demand System. In: Hong Kong Intl. Computer Conference, pp. 1–5 (1997)
3. Choe, Y.R., Pai, V.S.: Achieving Reliable Parallel Performance in a VoD Storage Server Using Randomization and Replication. In: Intl. Parallel and Distributed Processing Symposium, pp. 1–10 (2007)
4. Golubchik, L., Muntz, R.R., Chou, C.-F., Berso, S.: Design of Fault-Tolerant Large-Scale VOD Servers: With Emphasis on High-Performance and Low-Cost. IEEE Transactions on Parallel and Distributed Systems 12(4), 97–109 (2001)

5. Zhou, X., Xu, C.-Z.: Optimal Video Replication and Placement on a Cluster of Video-on-Demand Servers. In: Intl. Conference on Parallel Processing, pp. 547–555 (2002)
6. Huang, X.-M., Lin, C.-R., Chen, M.-S.: Design and Performance Study of Rate Staggering Storage for Scalable Video in a Disk-Array-Based Video Server. In: Intl. workshop on Network and Operating Systems support for Digital Audio and Video, pp. 177–182 (2005)

Position-Aware String Kernels with Weighted Shifts and a General Framework to Apply String Kernels to Other Structured Data

Kilho Shin

Carnegie Mellon CyLab Japan

Abstract. In combination with efficient kernel-base learning machines such as Support Vector Machine (SVM), string kernels have proven to be significantly effective in a wide range of research areas (*e.g.* bioinformatics, text analysis, voice analysis). Many of the string kernels proposed so far take advantage of simpler kernels such as trivial comparison of characters and/or substrings, and are classified into two classes: the *position-aware* string kernel which takes advantage of positional information of characters/substrings in their parent strings, and the *position-unaware* string kernel which does not. Although the positive semidefiniteness of kernels is a critical prerequisite for learning machines to work properly, a little has been known about the positive semidefiniteness of the position-aware string kernel. The present paper is the first paper that presents easily checkable sufficient conditions for the positive semidefiniteness of a certain useful subclass of the position-aware string kernel: the similarity/matching of pairs of characters/substrings is evaluated with weights determined according to *shifts* (the differences in the positions of characters/substrings). Such string kernels have been studied in the literature but insufficiently. In addition, by presenting a general framework for converting positive semidefinite string kernels into those for richer data structures such as trees and graphs, we generalize our results.

1 Introduction

The string kernel, in combination with efficient kernel-base learning machines such as Support Vector Machine (SVM), has been applied to a wide range of research areas (*e.g.* bioinformatics ([1,2,3]), text analysis ([4]), voice recognition), and proves to be significantly effective for clustering string-type data.

Many of the known string kernels were engineered based on simpler kernels such as trivial comparison of characters and/or substrings, and therefore are classified into two classes: the *position-aware* string kernel which takes advantage of positional information of characters/substrings in their parent strings, and the *position-unaware* string kernel which does not (*e.g.* the spectrum kernel ([1]), the string subsequence kernel ([4])). Some of position-aware string kernels evaluate only those pairs of characters/substrings whose positions in their parent strings coincide with each other (*e.g.* the locality-improved kernel ([2]) and the weighted-degree kernel ([3])). This constraint, however, is too restrictive for

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 316–325, 2007.

some applications, and it may be desirable to allow *weighted shifts* to improve generality — a pair of characters/substrings whose positions differ by a shift s is also evaluated but with a weight $\bar{w}_s$ determined according to s. The codon-improved kernel ([2]) and the weighed-degree kernel with shifts ([3]) are examples of kernels of this type. A brief survey of string kernels is given in Section 2.

Theoretically, string kernels with weighted shifts are expected to be effective when applied to string-type data such that discriminative patterns intensively distribute around plural particular positions (*e.g.* written/spoken texts, html documents). On the other hand we have to pay careful attention when engineering kernels of the type, since it is known that naïve selection of weights could easily harm the positive semidefiniteness of the resulting kernels. A kernel $K(x, y)$ is said to be positive semidefinite, if, and only if, for an arbitrary set of data $\{x_1, \ldots, x_N\}$, the derived Gram matrix $[K(x_i, x_j)]_{(i,j)\in\{1,\ldots,N\}^2}$ is positive semidefinite (*i.e.* all the eigenvalues are non-negative), and the positive semidefiniteness of a kernel is a critical prerequisite for kernel-base classifiers to work properly.

The present paper is the first paper that presents easily checkable sufficient conditions for position-aware string kernels with weighted shifts to be positive semidefinite (see Section 3 for the conditions, and Section 5 and Section 6 for proofs). A limited part of the results of the present paper is presented in [5] without proofs.

Furthermore, we describe a general framework for transforming positive semidefinite string kernels into positive semidefinite kernels for richer data structures such as trees and graphs (Section 4)

2 A Survey of String Kernels

In this section, we review 6 important string kernels in the literature, namely the spectrum kernel ([1]), the string subsequence kernel ([4]), the locality-improved kernel ([2]), the weighted-degree kernel ([3]), the codon-improved kernel ([2]), and the weighted-degree kernel with shifts ([3]). The spectrum kernel and the string subsequence kernel are examples of position-unaware kernels, while the others are examples of position-aware kernels. In particular, the codon-improved kernel and the shifted-weighted-degree kernel allow weighted sifts (see Section 1). Also, a good survey of kernels for structured data is available in [6].

In this section, we use the following notations. We define $C_{n,m}$ and $D_{n,m}$ as follows for integers $n \leq m$.

$$C_{n,m} = \{(i, i+1, \ldots, i+n-1) \mid 1 \leq i \leq m-n+1\}$$
$$D_{n,m} = \{(i_1, \ldots, i_n) \mid 1 \leq i_1 < \cdots < i_n \leq m\}$$

Strings are defined over an alphabet $\mathcal{A}$, and, for a string $x = x_1 x_2 \ldots x_{|x|}$ and a vector $\boldsymbol{i} \in D_{n,|x|}$, we let $x[\boldsymbol{i}]$ denote the n-length string $x_{i_1} x_{i_2} \ldots x_{i_n}$. The function $\delta(a, b)$ indicates Kronecker's delta function — it returns 1 if a and b are "identical", and 0 otherwise.

2.1 Position-Unaware String Kernels

Fix an integer $n \geq 1$ in this subsection. The spectrum kernel ([1]) was introduced for protein classification, and counts up the contiguous substrings of length n shared between input strings x and y.

$$K(x,y) = \sum_{\boldsymbol{i} \in C_{n,|x|}} \sum_{\boldsymbol{j} \in C_{n,|y|}} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$$

On the other hand, the string subsequence kernel ([4]) has proven to be effective for text classification. Compared with the spectrum kernel, the restriction on substrings with respect to contiguity is relaxed, and instead, the decay factor $\lambda \in (0,1)$ is introduced to emphasize contiguous substrings.

$$K(x,y) = \sum_{\boldsymbol{i} \in D_{n,|x|}} \sum_{\boldsymbol{j} \in D_{n,|y|}} \lambda^{i_n - i_1 + j_n - j_1} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$$

Both kernels only evaluate matching of substrings as sequences of characters, and don't take the positions of substrings into account at all.

2.2 Position-Aware String Kernels with Precise Position Matching

In some applications, it has proven true that positional information of substrings is an important factor to improve generality of machine learning.

The locality-improved kernel ([2]) is a successful example of string kernels exploiting positional information, and compares between nucleotide sequences x and y in order to recognize translation initiation sites (TIS). The locality-improved kernel $K(x,y)$ inspects matching between nucleotides at the same position in x and y, and is defined as follows with non-negative weights $\gamma_{|k|}$.

$$\mathrm{win}_p(x,y) = \left(\sum_{k=-\ell}^{\ell} \gamma_{|k|} \delta(x_{p+k}, y_{p+k}) \right)^{d_1}, \quad K(x,y) = \left(\sum_{p=\ell+1}^{L-\ell} \mathrm{win}_p(x,y) \right)^{d_2}$$

The weighted degree kernel ([3]) is another successful example of position-aware kernels, and is defined as follows.

$$K(x,y) = \sum_{n=1}^{d} \beta_n \sum_{\boldsymbol{i} \in C_{n,L}} \delta(x[\boldsymbol{i}], y[\boldsymbol{i}])$$

The kernels are different from each other in that the former performs character-by-character comparison, while the latter compares substrings. They, however are the same in that the character/substring pairs to be compared with each other are located at the same position in their parent strings.

2.3 Position-Aware String Kernels with Weighted Shifts

The codon-improved kernel ([2]) is a modification of the locality-improved kernel so as to exploit the *a priori* knowledge "a coding sequence (CDS) shifted by three nucleotides still looks like CDS ([2])". In fact, in addition to the matches of nucleotides placed at the same position, it compares the pairs of nucleotides whose positions differ exactly by 3. When T_3 denotes the 3-shift operator that chops off the leading 3 nucleotides, the window score $\mathrm{win}_p(x, y)$ at position p of the locality-improved kernel is modified as follows.

$$k_p(x, y) = \sum_{k=-\ell}^{\ell} \gamma_{|k|} \delta(x_{p+k}, y_{p+k})$$

$$\mathrm{win}_p(x, y) = \left[k_p(x, y) + \bar{w}\left\{k_p(T_3 x, y) + k_p(x, T_3 y)\right\}\right]^{d_1}$$

Although it is claimed in [2] that the codon-improved kernel is unconditionally positive semidefinite, the fact is that the weights should be chosen carefully. For simplicity, assume $\gamma_{|k|} = 1$, $d_1 = 1$, $\ell = 3q$ and $p = 3q + 1$. When x and y are the strings of length $6q + 1$ defined as follows, $\mathrm{win}_p(x, x) = \mathrm{win}_p(y, y) = 6q + 1$ and $\mathrm{win}_p(x, y) = 4q + 4\bar{w}q$ hold.

$$x = \underbrace{\texttt{ATGCGT ATGCGT...ATGCGT}}_{6q}\texttt{A}, \quad y = \underbrace{\texttt{CTGAGT CTGAGT...CTGAGT}}_{6q}\texttt{C}$$

Therefore, the determinant of the corresponding Gram matrix is $(1 - 2(2\bar{w} - 1)q)(1 + 2(2\bar{w} + 5)q)$, and is not always non-negative for $\bar{w} > \frac{1}{2}$.

In [3], the weighted degree kernel is modified along the same line as the codon-improved kernel except that the modified kernel includes $\bar{w}_s$ for plural s.

Let $k(x, y) = \sum_{n=1}^{d} \beta_n \sum_{\boldsymbol{i} \in C_{n,L}} w_{i_1} \delta(x[\boldsymbol{i}], y[\boldsymbol{i}])$. When T_s denotes the s-shift operator, the following kernel is introduced in [3][1].

$$K(x, y) = k(x, y) + \sum_{s=1}^{S} \bar{w}_s \{k(T_s x, y) + k(x, T_s y)\} \tag{1}$$

The positive semidefiniteness of $K(x, y)$ was investigated as follows in [3]. Since $k(x, y)$ is positive semidefinite, so is $k(x, y) + k(T_s x, y) + k(x, T_s y) + k(T_s x, T_s y)$. If w_i remain constant irrespective of i[2], it follows that $2k(x, y) + k(T_s x, y) + k(x, T_s y)$ is positive semidefinite. Therefore, $\sum_{s=1}^{S} 2\bar{w}_s \leq 1$ is a sufficient condition for $K(x, y)$ to be positive semidefinite. We can relax the constraint of $w_1 = \cdots = w_L$ to $w_1 \leq w_2 \leq \cdots \leq w_L$: The key property of $2k(x, y) + k(T_s x, y) + k(x, T_s y)$ remains true, since $k(x, y) - k(T_s x, T_s y) = \sum_{n=1}^{d} \beta_n \sum_{\boldsymbol{i} \in C_{n,L}} (w_{i_1} - w_{i_1 - s}) \delta(x[\boldsymbol{i}], y[\boldsymbol{i}])$ and $w_{i_1} - w_{i_1 - s} \geq 0$ hold with $w_i = 0$ for $i \leq 0$.

[1] Although S varies according to n in [3], we assume that S is a constant just for simplicity.

[2] This seems to be assumed in [3] with no declaration.

3 Our Contributions

As seen in the previous section, many existing string kernels are based on evaluation of comparison between characters/substrings of input strings. Also, to improve the performance of such character/substring-base string kernels, exploitation of the positional information of the characters/substrings is important, and, in fact, has proven to be effective at least in certain applications.

On the other hand, positive semidefiniteness of kernels, in principle, must be guaranteed, since kernel-based learning machines may not treat them properly, otherwise. In contrast to the position-aware string kernel which requires precise matching of the positions of characters/substrings (2.2), the positive semidefiniteness of the position-aware kernel with weighted shifts (2.3) is subtly affected by choice of weights, and, yet worse, only a little has been known about conditions on *good* weights.

The first contribution of this paper is to give an answer to the problem. In the remaining of this paper, without loss of generality, we assume that any strings are of length L over an alphabet $\mathcal{A}$. Given positional weights w_i, a positive shift s, a shift weight $\bar{w}_s$ and a kernel k over $\mathcal{A}$, we define $K(x, y)$ as follows.

$$K(x,y) = \sum_{i=1}^{L} w_i \left[k(x_i, y_i) + \bar{w}_s \left\{k(x_{i+s}, y_i) + k(x_i, y_{i+s})\right\}\right] \tag{2}$$

For non-negative integers a and b such that $a \in \{1, \ldots, s\}$ and $s(b-1)+a \leq L$, we define $\gamma_b^{(a)}$ by the recurrence formulas described below.

$$\gamma_0^{(a)} = 1, \quad \gamma_1^{(a)} = w_a, \quad \gamma_b^{(a)} = w_{s(b-1)+a}\gamma_{b-1}^{(a)} - \bar{w}_s^2 w_{s(b-2)+a}^2 \gamma_{b-2}^{(a)} \tag{3}$$

Then, our main theorem is stated as follows, and its proof is given in Section 5.

Theorem 1. *If $\gamma_b^{(a)} > 0$ holds for every (a, b) such that $a \in \{1, \ldots, s\}$ and $s(b-1)+a \leq L$, the character-base string kernel $K(x, y)$ defined by Eq. (2) is positive semidefinite for an arbitrary positive semidefinite kernel $k(x_i, y_j)$.*

Conversely, if $\gamma_b^{(a)} < 0$ holds for some (a, b), there exists a positive semidefinite kernel $k(x_i, y_j)$ such that the resulting $K(x, y)$ is not positive semidefinite.

The sufficient condition presented in Theorem 1 is *very close* to a necessary condition, since the positive semidefiniteness of $K(x, y)$ is left undetermined only in the marginal cases where $\gamma_b^{(a)} \geq 0$ for all (a, b) and $\gamma_b^{(a)} = 0$ for some (a, b).

On the other hand, when $w_1, \ldots, w_L$ are fixed, the condition is reduced to an equivalent inequality of $0 \leq \bar{w}_s < b_{w_1,\ldots,w_L}^{(s)}$ for some $b_{w_1,\ldots,w_L}^{(s)}$. While it is not easy to determine the actual values for $b_{w_1,\ldots,w_L}^{(s)}$, Corollary 1 gives an easily computable lower bound for $b_{w_1,\ldots,w_L}^{(s)}$.

Corollary 1. *Assume that all the weights are positive. The kernel K defined by Eq.(2) is positive semidefinite for an arbitrary positive semidefinite $k(x_i, y_j)$, if the following inequality holds for $w_1, \ldots, w_L$ and $\bar{w}_s$.*

$$\bar{w}_s \leq \min \left\{ \frac{w_i}{w_{i-s} + w_i} \middle| i = s+1, \ldots, L \right\} \tag{4}$$

Now, let us consider the kernel of the following form. In the same way as in the above, $k(x_i, y_j)$ is a positive semidefinite kernel over $\mathcal{A}$.

$$K(x, y) = \sum_{i=1}^{L} w_i \left[k(x_i, y_i) + \sum_{s=1}^{S} \bar{w}_s \left\{ k(x_{i+s}, y_i) + k(x_i, y_{i+s}) \right\} \right] \tag{5}$$

Let $b'^{(s)}_{w_1,\ldots,w_L}$ be positive numbers such that, if $0 \leq \bar{w}_s \leq b'^{(s)}_{w_1,\ldots,w_L}$, the kernel of Eq. (2) is positive semidefinite. If we have $\sum_{s=1}^{S} \alpha_s = 1$ such that $0 \leq \bar{w}_s \leq \alpha_s b'^{(s)}_{w_1,\ldots,w_L}$, $K_s(x, y)$ defined below is positive semidefinite, and therefore so is $K(x, y) = \sum_{s=1}^{S} K_s(x, y)$.

$$K_s(x, y) = \sum_{i=1}^{L} w_i \left[\alpha_s k(x_i, y_i) + \bar{w}_s \left\{ k(x_{i+s}, y_i) + k(x_i, y_{i+s}) \right\} \right]$$

Thus, we have obtained Theorem 2.

Theorem 2. *If the following inequality holds for $\bar{w}_s$, the character-base string kernel of Eq. (5) is positive semidefinite for an arbitrary positive semidefinite $k(x_i, y_j)$.*

$$\sum_{s=1}^{S} \frac{\bar{w}_s}{b'^{(s)}_{w_1,\ldots,w_L}} \leq 1$$

Proof. We have only to take α_s such that $\frac{\bar{w}_s}{b'^{(s)}_{w_1,\ldots,w_L}} \leq \alpha_s$ and $\sum_s^S \alpha_s = 1$. □

The sufficient condition by [3], which was also described in 2.3, is obtained as a corollary to Corollary 1 and Theorem 2.

Corollary 2. *If $w_1 \leq \cdots \leq w_L$, the character-base string kernel of Eq. (5) with $\sum_{s=1}^{S} \bar{w}_s \leq \frac{1}{2}$ is positive semidefinite.*

In Section 4, we introduce a general framework to transform character-base string kernels into not only substring-base string kernels but also kernels for richer data structures than strings. Here, with the framework, we derive from Eq. (5) two types of position-aware substring-base string kernels with weighted shifts: one is the weighted-degree kernel with shift described in 2.3 (Eq.(1) and [3]), and the other is its variation for non-contiguous substrings (Eq.(6)).

4 A Framework for Transforming Character-Base String Kernels into Kernels over Other Structured Data

In this section, we present a general framework to transform given *character-base* string kernels into not only *substring-base* string kernels but also kernels for richer data structures such as trees and graphs.

We start with defining the framework in a formal manner, and then look closely at it using examples. Let $\chi, \chi', \{\chi'_x \mid x \in \chi\}, \mu$ and k' be as follows.

- χ is a space of data points.
- χ' is a space of *subparts* (*e.g.* characters, substrings, subtrees, subgraphs)
- Per each $x \in \chi$, a finite set $\chi'_x \subseteq \chi'$ is assigned.
- $\mu : \chi' \to \mathbb{N}$ is a positioning mapping. Further, we denote $\max \mu(\chi'_x)$ by $|x|$.
- $k' : \chi' \times \chi' \to \mathbb{R}$ is a positive semidefinite kernel.

Given $(\chi, \chi', \{\chi'_x\}, \mu, k')$, we define an alphabet $\mathfrak{A}$, a mapping $\mathfrak{L} : \chi \to \mathfrak{A}^*$ and a kernel $k : \mathfrak{A} \times \mathfrak{A} \to \mathbb{R}$ as follows.

- The alphabet $\mathfrak{A}$ is the power set $\mathfrak{P}(\chi')$ of χ'.
- For $x \in \chi$, the *lift* of x, denoted by $\mathfrak{L}(x)$, is the string of length $|x|$ whose i-th character $\mathfrak{L}(x)_i$ is $\{x' \in \chi'_x \mid \mu(x') = i\} \in \mathfrak{A}$.
- A kernel $k : \mathfrak{A} \times \mathfrak{A} \to \mathbb{R}$ is defined by $k(X, Y) = \sum_{x' \in X} \sum_{y' \in Y} k'(x', y')$.

Let $K(\xi, \eta)$ be an arbitrary character-base string kernel, which includes a *character kernel* $k(\xi_i, \eta_j)$. Furthermore, we assume that $K(\xi, \eta)$ has the property that it is positive semidefinite, if so is $k(\xi_i, \eta_j)$ (as Theorem 1, Corollary 1 and Theorem 2 assert). We define $K(x, y)$ for $x, y \in \chi$ by substituting $k(\mathfrak{L}(x)_i, \mathfrak{L}(y)_j)$ for $k(\xi_i, \eta_j)$ in $K(\xi, \eta)$. Since Haussler's theorem ([7]) asserts that $k(\mathfrak{L}(x)_i, \mathfrak{L}(y)_j)$ is positive semidefinite, $K(x, y)$ remains positive semidefinite.

With this framework, the substring-base string kernels described in Section 2, namely the spectrum kernel (Sp), the string subsequence kernel (SSs), the weighted-degree kernel (WD), the weighted-degree kernel with shifts (WDwS) and the non-contiguous substring version of WDwS (Eq. (6)), are all derived from some of the character-base string kernels that we discuss in the present paper. Table 1 describes the necessary settings for the derivation. In particular, Theorem 1, Corollary 1 and Theorem 2 provide sufficient conditions on the weights for $K(x, y)$ of WDwS and Eq. (6) to be positive semidefinite.

$$\bar{w}_0 = 0,\ K(x, y) = \sum_{\boldsymbol{i} \in D_{n,L}} \sum_{\boldsymbol{j} \in D_{n,L}} \sum_{s=0}^{S} w_{\min\{i_1, j_1\}} \bar{w}_s \delta(|i_1 - j_1|, s) \delta(x[\boldsymbol{i}], y[\boldsymbol{j}]) \quad (6)$$

Furthermore, we can apply the framework to structured data other than strings. For example, let χ be a set of rooted trees, and let χ'_x denote the set of the subtrees of x. When the *depth* of a vertex v of a tree x is defined as the number of edges of the upward path from v to the root of x, we define $\mathsf{dpth}(x')$ for $x' \in \chi'_x$ as the depth of the root of x' in x. Then, with the setting

Table 1. Settings for applying the framework to respective string kernels

Kernel	$\chi'_x =$	$\mu((\boldsymbol{i}, x)) =$	$k'((\boldsymbol{i}, x), (\boldsymbol{j}, y)) =$	Char.-base kernel
Sp	$C_{n,\lvert x\rvert} \times \{x\}$	i_1	$\delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	$\sum_{i=1}^{\lvert\xi\rvert} \sum_{j=1}^{\lvert\eta\rvert} k(\xi_i, \eta_j)$
SSs	$D_{n,\lvert x\rvert} \times \{x\}$	i_1	$\lambda^{i_n - i_1 + j_n - j_1} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	$\sum_{i=1}^{\lvert\xi\rvert} \sum_{j=1}^{\lvert\eta\rvert} k(\xi_i, \eta_j)$
WD	$\left(\bigcup_{n=1}^{d} C_{n,\lvert x\rvert}\right) \times \{x\}$	i_1	$\beta_{\lvert\boldsymbol{i}\rvert} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	$\sum_{i=1}^{\min\{\lvert x\rvert, \lvert y\rvert\}} k(\xi_i, \eta_i)$
WDwS	$\left(\bigcup_{n=1}^{d} C_{n,\lvert x\rvert}\right) \times \{x\}$	i_1	$\beta_{\lvert\boldsymbol{i}\rvert} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	Eq. (5)
Eq. (6)	$\left(\bigcup_{n=1}^{d} D_{n,\lvert x\rvert}\right) \times \{x\}$	i_1	$\beta_{\lvert\boldsymbol{i}\rvert} \delta(x[\boldsymbol{i}], y[\boldsymbol{j}])$	Eq. (5)

of $\mu(x') = \mathsf{dpth}(x')$ and $k'(x', y') = \delta(x', y')$, we will obtain from the character-base string kernel of Eq. (5) a tree kernel that counts isomorphic subtree pairs $(x', y') \in \chi'_x \times \chi'_y$ with the weights $w_{\min\{\mathsf{dpth}(x'), \mathsf{dpth}(y')\}} \cdot \bar{w}_{\lvert \mathsf{dpth}(x') - \mathsf{dpth}(y')\rvert}$.

$$K(x, y) = \sum_{x' \in \chi'_x} \sum_{y' \in \chi'_y} \sum_{s=0}^{S} w_{\min\{\mathsf{dpth}(x'), \mathsf{dpth}(y')\}} \bar{w}_s \delta(\lvert \mathsf{dpth}(x') - \mathsf{dpth}(y')\rvert, s) \delta(x', y)$$

This tree kernel would be useful to classify web page trees, where the distance of a page from its root page has significance. Also, we can use the order $\mathsf{preodr}(x')$ derived from the pre-order traversal of trees instead of the depth $\mathsf{dpth}(x')$. The resulting tree kernel would be useful to classify parse trees of natural languages, for example, where the word order in sentences has important meaning.

5 Proof of Theorem 1

Here, we will take advantage of the result of [5]. Indeed, our key lemma, namely, Lemma 1, is a degenerated corollary to Theorem 1 in [5], which gives a general sufficient condition of multivariate polynomials of arbitrary degrees such that *polynomial kernels* derived from the polynomials become positive semidefinite.

Lemma 1 ([5]). *Let $\mathcal{A}$ be an alphabet, and let x_i denote the i-th character of an L-length string $x \in \mathcal{A}^L$. For an L-dimensional real matrix $C = [c_{i,j}]_{(i,j) \in \{1,\ldots,L\}^2}$, the following are equivalent to each other.*

1. *C is positive semidefinite.*
2. *For an arbitrary positive semidefinite $k : \mathcal{A} \times \mathcal{A} \to \mathbb{R}$, the kernel that returns $\sum_{i=1}^{L} \sum_{j=1}^{L} c_{i,j} k(x_i, y_j)$ on input of $(x, y) \in \mathcal{A}^L \times \mathcal{A}^L$ is also positive semidefinite.*

When we define $c_{i,j}$ by: $c_{i,j} = w_i$, if $i = j$; $c_{i,j} = \bar{w}_s w_i$, if $j = i + s$; $c_{i,j} = \bar{w}_s w_j$, if $i = j + s$; and $c_{i,j} = 0$, otherwise. Then, $K(x, y) = \sum_{i=1}^{L} \sum_{j=1}^{L} c_{i,j} k(x_i, y_j)$ holds for $K(x, y)$ of Eq.(2). Therefore, Lemma 1 implies that, to prove the first assertion of Theorem 1, it suffices to show that the matrix $C = [c_{i,j}]_{(i,j) \in \{1,\ldots,L\}^2}$ is positive semidefinite.

We let a be one of $\{1, 2, \ldots, s\}$, and let the submatrix $C_b^{(a)}$ denote the b-dimensional matrix $[c_{s(i-1)+a,s(j-1)+a}]_{(i,j)=\{1,\ldots,b\}^2}$. For $L = sq + r$ such that $r \in \{1, 2, \ldots, s\}$, b moves in the interval $[1, q + 1]$ if $a \leq r$, and does in the interval $[1, q]$ if $a > r$. For example, when $b = 4$, $C_4^{(a)}$ looks as follows.

$$C_4^{(a)} = \begin{bmatrix} w_a & \bar{w}_s w_a & 0 & 0 \\ \bar{w}_s w_a & w_{s+a} & \bar{w}_s w_{s+a} & 0 \\ 0 & \bar{w}_s w_{s+a} & w_{2s+a} & \bar{w}_s w_{2s+a} \\ 0 & 0 & \bar{w}_s w_{2s+a} & w_{3s+a} \end{bmatrix}$$

By applying the same permutation to the rows and the columns of C if necessary, C is decomposed into a direct sum of its sub-matrices as follows.

$$C = C_{q+1}^{(1)} \oplus \cdots \oplus C_{q+1}^{(r)} \oplus C_q^{(r+1)} \oplus \cdots \oplus C_q^{(s)}$$

Therefore, C is positive definite (*i.e.* C has only positive eigenvalues), if, and only if, so are $C_{q+1}^{(a)}$ for $a \leq r$ and $C_q^{(a)}$ for $a > r$.

On the other hand, $C_{q+1}^{(a)}$ for $a \leq r$ (resp. $C_q^{(a)}$ for $a > r$) is positive definite, if, and only if, $\det(C_b^{(a)}) > 0$ for all $1 \leq b \leq q + 1$ (resp. $1 \leq b \leq q$)[3]. Since, by the Laplacian determinant expansion by minors, we have the recurrence formula (7) for $\det(C_0^{(a)}) = 1$ and $\det(C_1^{(a)}) = w_a$. This indicates that $\det(C_b^{(a)})$ coincides with $\gamma_b^{(a)}$. Thus, the first assertion of Theorem 1 has been proved.

$$\det(C_b^{(a)}) = w_{(b-1)s+a} \det(C_{b-1}^{(a)}) - (w_{(b-2)s+a}\bar{w}_s)^2 \det(C_{b-2}^{(a)}) \tag{7}$$

The second assertion is also derived from Lemma 1. If $\gamma_b^{(a)} < 0$ for some (a, b), $\det(C)$ is negative, and hence C is not positive semidefinite. By Lemma 1, there exists a positive semidefinite kernel $k(x_i, y_j)$ defined over the alphabet $\mathcal{A}$ such that $K(x, y)$ is not positive semidefinite.

6 Proof of Corollary 1

In this section, assuming that $\bar{w}_s$ satisfies the inequality (4), we will prove that $\gamma_b^{(a)} > 0$ holds for an arbitrary pair of non-negative integers (a, b) such that $a \in \{1, \ldots, s\}$ and $s(b - 1) + a \leq L$.

To start with, we define $B_b^{(a)}$ as the matrix obtained by replacing the (b, b)-element $w_{s(b-1)+a}$ of $C_b^{(a)}$ with $\bar{w}_s w_{s(b-1)+a}$, and let $\beta_b^{(a)} = \det(B_b^{(a)})$. For example, $B_4^{(a)}$ looks as follows (compare with $C_4^{(a)}$).

$$B_4^{(a)} = \begin{bmatrix} w_a & \bar{w}_s w_a & 0 & 0 \\ \bar{w}_s w_a & w_{s+a} & \bar{w}_s w_{s+a} & 0 \\ 0 & \bar{w}_s w_{s+a} & w_{2s+a} & \bar{w}_s w_{2s+a} \\ 0 & 0 & \bar{w}_s w_{2s+a} & \bar{w}_s w_{3s+a} \end{bmatrix}$$

[3] A symmetric real matrix $[a_{i,j}]_{(i,j)\in\{1,\ldots n\}^2}$ is positive definite, if, and only if, $\det([a_{i,j}]_{(i,j)\in\{1,\ldots m\}^2}) > 0$ for all $1 \leq m \leq n$. It is easy to prove it by induction on m. Also, the reader may refer to [8] for a proof.

In the rest of this section, we fix $a \in \{1, \ldots, s\}$, and prove $\gamma_b^{(a)} > 0$ and $\beta_b^{(a)} > 0$ by induction on b. Furthermore, we can assume $b > 1$, since $\gamma_1^{(a)} = w_a > 0$ and $\beta_1^{(a)} = \bar{w}_s w_a > 0$ hold.

First, we confirm a few key properties.

- The hypothesis (4) implies $\bar{w}_s < 1$.
- Therefore, $\gamma_b^{(a)} > \beta_b^{(a)}$ follows from the hypothesis of induction $\gamma_{b-1}^{(a)} > 0$. This implies that we have only to show $\beta_b^{(a)} > 0$ to complete the proof.
- The inequality $\bar{w}_s \leq w_{s(b-1)+a}/(w_{s(b-2)+a} + w_{s(b-1)+a})$ implies the following.

$$1 - \frac{\bar{w}_s w_{s(b-2)+a}}{w_{s(b-1)+a}} \geq 1 - \bar{w}_s \left(\frac{1}{\bar{w}_s} - 1 \right) = \bar{w}_s \tag{8}$$

To show $\beta_b^{(a)} > 0$, we first expand $\beta_b^{(a)}$ and $\gamma_{b-1}^{(a)}$ by Laplacian determinant expansion, apply the inequality of (8) (note that $\gamma_{b-2}^{(a)} > 0$ holds by the hypothesis of induction), and then collect up the terms into $\beta_{b-1}^{(a)}$ by applying Laplacian determinant expansion in reverse. The assertion follows from the hypothesis of induction $\beta_{b-1}^{(a)} > 0$.

$$\begin{aligned}
\beta_b^{(a)} &= \bar{w}_s w_{s(b-1)+a} \gamma_{b-1}^{(a)} - (\bar{w}_s w_{s(b-2)+a})^2 \gamma_{b-2}^{(a)} \\
&= \bar{w}_s w_{s(b-1)+a} \left\{ \left(1 - \frac{\bar{w}_s w_{s(b-2)+a}}{w_{s(b-1)+a}} \right) w_{s(b-2)+a} \gamma_{b-2}^{(a)} - \bar{w}_s^2 w_{s(b-3)+a}^2 \gamma_{b-3}^{(a)} \right\} \\
&\geq \bar{w}_s w_{s(b-1)+a} \left(\bar{w}_s w_{s(b-2)+a} \gamma_{b-2}^{(a)} - \bar{w}_s^2 w_{s(b-3)+a}^2 \gamma_{b-3}^{(a)} \right) \\
&= \bar{w}_s w_{s(b-1)+a} \beta_{b-1}^{(a)}
\end{aligned}$$

References

1. Leslie, C., Eskin, E., Noble, W.: The spectrum kernel: a string kernel for svm protein classification. In: 7th Pacific Symposium of Biocomputing (2002)
2. Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T., Müller, K.R.: Engineering support vector machne kernels that recognize translation initiation sites. Bioinformatics 16, 799–807 (2000)
3. Rätsch, G., Sonnenburg, S., Schölkopf, B.: Rase: recognition of alternatively spliced exons in c.elegans. Bioinformatics 21, 369–377 (2005)
4. Lodhi, H., Shawe-Taylor, J., Cristianini, N., Watkins, C.J.C.H.: Text classificatio using string kernels. Advances in Neural Information Processing Systems 13 (2001)
5. Shin, K., Kuboyama, T.: Polynomial summaries of positive semidefinite kernels. In: ALT 2007. The 18th International Conference on Algorithmic Learning Theory (to appear)
6. Gärtner, T.: A survey of kernels for structured data. SIGKDD Explorations 5, 49–58 (2003)
7. Haussler, D.: Convolution kernels on discrete structures. UCSC-CRL 99-10, Dept. of Computer Science, University of California at Santa Cruz (1999)
8. Berg, C., Christensen, J.P.R., Ressel, R.: Harmonic Analysis on semigroups. Theory of positive definite and related functions. Springer, Heidelberg (1984)

A New Regression Based Software Cost Estimation Model Using Power Values

Oktay Adalier[1], Aybars Uğur[2], Serdar Korukoğlu[2], and Kadir Ertaş[3]

[1] TUBITAK-UEKAE, National Research Institute of Electronics and Cryptology, PK 74 Gebze KOCAELI - Turkey
Oadalier@uekae.tubitak.gov.tr
[2] Ege University, Department of Computer Engineering, Bornova IZMIR – Turkey
{aybars.ugur, serdar.korukoglu}@ege.edu.tr
[3] Dokuz Eylül University, Department of Econometrics, Buca, IZMIR – Turkey
kadir.ertas@deu.edu.tr

Abstract. The paper aims to provide for the improvement of software estimation research through a new regression model. The study design of the paper is organized as follows. Evaluation of estimation methods based on historical data sets requires that these data sets be representative for current or future projects. For that reason the data set for software cost estimation model the International Software Benchmarking Standards Group (ISBSG) data set Release 9 is used. The data set records true project values in the real world, and can be used to extract information to predict new projects cost in terms of effort. As estimation method regression models are used. The main contribution of this study is the new cost production function that is used to obtain software cost estimation. The new proposed cost estimation function performance is compared with related work in the literature. In the study same calibration on the production function is made in order to obtain maximum performance. There is some important discussion on how the results can be improved and how they can be applied to other estimation models and datasets.

Keywords: Software Cost Estimation, Regression Analysis, Software Cost Models.

1 Introduction

Software development effort estimates are the basis for project bidding, budgeting and planning. These are critical practices in the software industry, because poor budgeting and planning often has dramatic consequences [1]. The common argument on the project cost overruns is very large. Boraso reported that [2] 60% of large projects significantly overrun their estimates and 15% of the software projects are never completed due to the gross misestimating of development effort.

Delivering a software product on time, within budget, and to an agreed level of quality is a critical concern for software organizations. Accurate estimates are crucial for better planning, monitoring and control [3]. Jones [4] proposes software quality as

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 326–334, 2007.

"software quality means being on time, within budget, and meeting user needs". On the other hand, it is necessary to give the customer or the developer organization an early indication of the project costs.

As a consequence, considerable research attention is now directed at gaining a better understanding of the software development process as well as constructing and evaluating software cost estimation tools. Therefore, there has been excessive focus of research on estimation methods from a variety of fields.

Statistical regression analysis is the most suitable technique to calibrate performance models in a black-box fashion. Statistical regression analysis is a technique which models the relation between a set of input variables, and one or more output variables, which are considered somewhat dependent on the inputs, on the basis of a finite set of input/output observations. The estimated model is essentially a predictor, which, once fed with a particular value of the input variables, returns a prediction of the value of the output. The goal is to obtain a reliable generalisation, that means that the predictor, calibrated on the basis of a finite set of observed measures, is able to return an accurate prediction of the dependent variable when a previously unseen value of the independent vector is presented. In other terms, this technique aims to discover and to assess, on the basis of observations only, potential correlations between sets of variables and use these correlations to extrapolate to new scenarios [5].

The paper is structured according to the Jorgensen's [6] software development estimation study classification. That is the research topic of the study is cost production function and its calibration. Estimation approach of the work is regression analysis and expert judgment. Research approach covers real-life evaluation and history-based evaluation data. The data set which is used in this study are obtained from the International Software Benchmarking Standards Group (ISBSG) data set Release 9. The data set records true project values in the real world, and can be used to extract information to predict new projects cost in terms of effort.

The rest of the paper is structured as follows: Section 2 starts with a discussion on related work. The evaluation criteria are presented in Section 3. Section 4 follows with the description of the data set and the data preparation. Section 5 explains the proposed estimation methods applied. Section 6 summarizes and discusses the results of the analysis. Finally, Section 7 presents the conclusions and discussion of practical implications.

2 Related Work

Regression-based estimation approaches dominate. Notice that regression-based estimation approaches include most common parametric estimation models. More than half of the estimation papers try to improve or compare with regression model based estimation methods. Statistical regression analysis is the most suitable technique to calibrate performance models in a black- box fashion [5]. The problem of cost modeling can be attacked within the framework of statistical regression. The idea is that statistical techniques can discover complex relations between input independent variables and output dependent variables. The estimated model is essentially a predictor, which once fed with a particular value of the input variables,

returns a prediction of the value of the output. In other terms, this technique aims to discover and to assess, on the basis of observations, potential correlations between sets of variables and use these correlations to extrapolate to new scenarios.

In our case, we are interested in building a regression model based on the training data to use it subsequently to predict the total effort in man-months of future software projects.

Wieczorek [3] make four different studies with different data sets. Especially her third and fourth study with ISBSG dataset is valuable for our study. It covers 145 and 324 projects respectively. In her study system size (function points) was identified as the most important cost driver by all methods and for all datasets. Her evaluation criteria was the magnitude of relative error and as prediction level Pred(0.25) is used. He used in her study ordinary least squares regression and stepwise analysis of variance. She found that the descriptive statistics for the variables Organization type, System size in terms of function point and productivity.

Liu [7] used ISBSG data set in her study. She tests the correlation between the explanatory numerical variables and the response variable by running a stepwise regression analysis. At the end of her study she obtains adjusted R^2 –value 0.6275. Therefore her model represents whole ISBSG dataset not more than 65%. Stensrud [8] uses a dataset with 48 completed projects extracted from an internal database in Andersen Consulting. He calculates the adjusted R^2 –value by 80.1%. Hu [9] reviews software cost models. He presents the Minimum Software Cost Model (MSCM), derived from economic production theory and system optimization. The MSCM model is compared with COCOMO, SLIM estimation models. As known COCOMO is an empirical model derived from statistical regression. SLIM has its roots in the Rayleigh manpower distribution. For comparative purpose, two classic production models, the generalized Cobb-Douglas production (GCD) and the generalized Cobb-Douglas production with time factor (GCDT) are also included. He note that MSCM model also takes the general form of the Cobb-Douglas production function except that it requires $\alpha + \beta = 1$. They used the magnitude of error (MRE) to measure the quality of estimation of each model. They obtain in their study the adjusted R^2 –value for MSCM 89%, for GCDT 56% and for GCD 54% respectively.

3 Evaluation Criteria

The validation of a model is also a persistent issue in the construction of software cost functions. Depending on the authors, different measures of the goodness of fit of the model have been considered. The coefficient of multiple determinations R^2, and adjusted R^2, obtained from regression analysis is being used as a measure of the capability of explanation of the model. Analysis of the residuals seems a necessary condition for examining the aptness of the model, and outlier examination has been suggested for examining the model stability. There is a wide consensus in using the magnitude of relative error (MRE) as an essential element for assessing a regression model [10].

The magnitude of relative error as a percentage of the actual effort for a project is defined as:

$$MRE = \left| \frac{Effort_{Actual} - Effort_{Estimated}}{Effort_{Actual}} \right| \quad (1)$$

The MRE is calculated for each project in the data sets.

In addition the prediction level Pred(r), prediction at level r, is also used. This measure is often used in the literature and is a proportion of observations for a given level of accuracy:

$$\Pr ed(r) = k / N \quad (2)$$

where, N is the total number of observations, and k the number of observations with an MRE less than or equal to r. Wieczorek [3] and Boraso [2] report that according to Conte, a value of Pred(0.25) ≥75% and $\overline{MRE}$ ≤25% are considered desirable for effort models. It seen that the accuracy of an estimation technique is proportional to the Pred(0.25) and inversely proportional to the MRE and the mean MRE, $\overline{MRE}$.

4 Data Set Description

Jorgensen state that the potential importance of researchers with a long-term focus on software cost estimation can be illustrated through an analysis of the papers covering the topics "measures of estimation performance" and "data set properties." He concludes these topics are basic research on cost estimation necessary for meaningful analyzes and evaluations of estimation methods [6].

He states that most of the publications evaluate an estimation method by applying historical data. Therefore, he believes that the lack of publications on real-life use of estimation methods point at a potentially important shortcoming. According to the author, if the estimation context were made with real-life professional project dataset then the realism of the estimation study will increase.

The study in this paper is based on the International Software Benchmarking Standards Group (ISBSG, Release 9) data [11]. It contains data for 3.024 projects all around the world. It should be noted that the data in the Repository has come from twenty countries, with 70% of the projects being less than six years old. This is what makes the ISBSG Repository unique. A broad range of project types from many industries and many business areas are available for our study to use for estimating, awareness of trends, comparison of platforms and languages or benchmarking. 57% are enhancement projects, 41% are new developments, and 2% are re-developments. The number of projects according to their development type is shown in Table 1.

Table 1. ISBSG R9 data set number according to four development types is given

Type of development	Projects
Enhancement	1711
New development	1246
Re-development	65
Other	2
Total	3024

The ISBSG project has developed and refined its data collection standard over ten year period based on the metrics that have proven to be most useful in helping to improve software development management and process. The ISBSG R9 data set records true project values in the real world, and can be used to extract information to predict new projects cost in terms of effort. In the data set the reliability of samples are rated as 4 levels from A to D where level 'A' represents the most reliable data. In ISBSG R9) data set, there are 734 'A' rated observations, each containing 50 explanatory variables (project size, time, programming language, etc.) and 2 response variables (Summary work effort and normalized work effort). After removing the observations that contain missing attribute values, from level 'A' data set, there are 112 project records remaining.

We choose normalized work effort as response variable or dependent variable. Each observation contains 4 variables 3 of which are the independent variables such as source lines of code (SLOC), adjusted function points, normalized productivity delivery rate. The primary motivation for choosing adjusted function points was the need to determine an estimate of system size and hence development effort early in the development lifecycle the same need is stated also by Finnie [12]. The SLOC is the number of the source lines of code produced by the project. Since this is only available for some projects in the data set. For this reason only 112 of the projects are chosen from the data set.

It is not a surprised that none of the chosen explanatory variables are normally distributed. Figure 1 shows the distribution of raw data of explanatory variables.

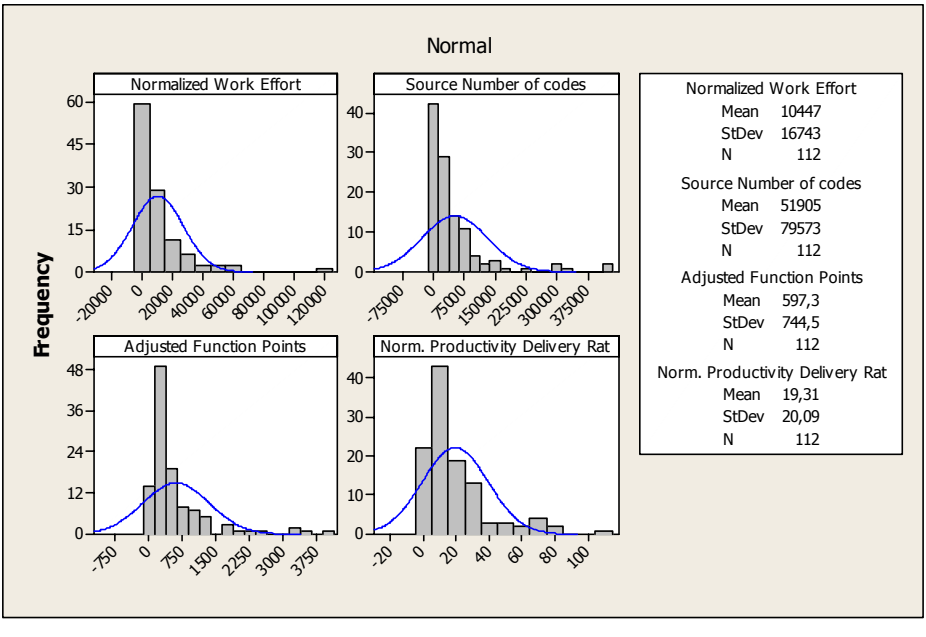


Fig. 1. Histogram of original raw data of explanatory variables

These variables are transformed to approximate a normal distribution. The natural logarithm to base e (natural logarithms) is applied to the data set. The correlation between explanatory variables is not high.

5 Proposed Model

Dolado [10] stated that the consumption of resources can be seen from two complementary viewpoints: as a cost function or as a production function. The effort is the input to the process and the output is the software product. This problem is the dual one of finding a cost function for the software product, although from the methodological point of view the issues are similar in both cases. Almost all works in the literature discuss the problem of project estimation from the perspective of the identification of the cost function, in which the problem under study is to identify the function which relates the size of the product to the man months employed. This is the approach followed here. The underlying idea for building the model is to consider that the software product is manufactured in some way, and the factors that affect the cost of the project is the size of the software which is given as source number of codes, adjusted function points and normalized productivity delivery rate. The equation is calibrated to fit to the actual data points of the projects which are given in the dataset. Since all the p-values are smaller then 0.01, there is sufficient evidence at alpha = 0.01 that the predictor variables are correlated with the response variable. As a result we can use the predictor variables source number of codes, adjusted function points and normalized productivity delivery rate to predict the effort of the project.

The use of the above stated predictor variables is supported by Delapy [13]. Which claims that one simple way of calculating the effect involved in the development of a software system requires a measure of size of the system (for example in lines of code or function points) and the expected productivity of the project team. Similarly, Shepperd [14] stated that the most straightforward method is to assuming a linear model which uses regression analysis to estimate the coefficients of the below shown equation.

$$y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 + \varepsilon \tag{3}$$

where x_1 is Source Number of codes, x_2 is Adjusted Function Points and x_3 is Normalized Productivity Delivery Rate. y is the estimated effort in terms of manpower in man-month. α_0, α_1, α_2, and α_3 are the coefficients to be estimated. ε is the error term parameter with a normal distribution.

Since, in the previous section we have transformed the data set with natural logarithm in order to normalize the distribution of the data. The effort production function became as follows:

$$y^{\frac{1}{\beta}} = \alpha_0 + \alpha_1 \log x_1 + \alpha_2 \log x_2 + \alpha_3 \log x_3 + \varepsilon \tag{4}$$

to expressed it more formally became following production function

$$y = (\alpha_0 + \sum_{i=1}^{k} \alpha_i \log x_i + \varepsilon)^{\beta} \tag{5}$$

where k=3, β = 2,3,…., 10.
the equation can be also be expressed to easy use as follows:

$$y = (\alpha_0 + \log(\prod_{i=1}^{k} x_i^{\alpha_i}) + \varepsilon)^{\beta} \tag{6}$$

We applied multiple linear regressions to the dataset. The regression model of MRE and adjusted R^2 were developed by applying stepwise regression. The resulting regression models, and connected significance values, from this process are listed in

Table 2. Regression Analysis results for different powered estimation functions

β	Coefficients $\alpha_0, \alpha_1, \alpha_2, \alpha_3$	MRE	R^2 –adjusted	PRED(0,25)
2	-240, -4.80, 45.3, 42.2	539,57	85,1%	71,4%
3	-30.5, -0.427, 6.40, 6.14	5,2589	92,8%	93,7%
4	-8.97, -0.109, 2.25, 2.19	0,3833	95,8%	96,4%
5	-3.57, -0.0433, 1.15, 1.13	0,0658	97,3%	99,1%
6	-1.56, -0.0219, 0.711, 0.700	0,0179	98,1%	>=99,1%
7	-0.616, -0.0128, 0.494, 0.487	0,0064	98,6%	>=99,1%
8	-0.107, –0.00829, 0.369, 0.365	0,0027	98,9%	>=99,1%
9	0.198, –0.00573, 0.290, 0.288	0,0013	99,1%	>=99,1%
10	0.393,–0.00416, 0.237, 0.235	0,0007	99,3%	>=99,1%

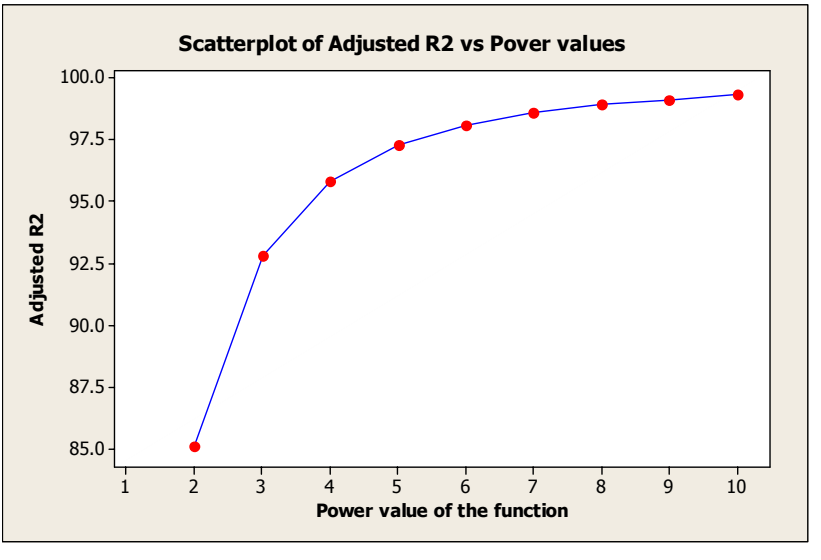


Fig. 2. Change of adjusted R2 values according to power degree of the equation

the Table 2. β is the power value for the effort estimation function (4). Coefficients are the estimated values for equation (4). MRE is the magnitude of relative error for each estimation function. In Table 2 it can be seen that the rate of MRE is decreasing when power is increasing. At power 5 we reach the required significant level of error rate. The adjusted R^2 is increasing very rapidly. In Figure 2 the change of adjusted R^2 can be followed. The accuracy of proposed model is very high because the Pred(0.25) values are very high and inversely the MRE values are very low.

Table 2 indicates the level of robustness of the variables included in the model. The variables included in our MRE-model are included in all best subset-models with number of variables greater than three, that is, the variable inclusion seems to be robust. The adjusted R^2 is satisfactory (85,1%), even for the model with the equation (4) who has power value of 2. This means that the majority of the variance in estimation accuracy is described by our model. A high adjusted R^2 entails that the model we found validates in explaining the relationship between the mean estimation accuracy and the included variables.

6 Comparison and Analysis of Results

We have compared our study with the study results of Liu [7], Stensrud [8] and Hu [9]. Hu uses Kamerer [9] data set for their analysis. He tries 5 different models and obtains best results for their calculations as adjusted R^2 89% for Minimum Software Cost Model (MSCM), adjusted R^2 56% for generalized Douglas production with time factor (GCDT) and adjusted R^2 54% for generalized Cobb-Douglas production (GCD). Liu uses the ISBSG data set as we have used and obtains an adjusted R^2 value as 61% in her study. Stensrud uses Laturi database as data set. He obtains MRE as %34 minimum value and PRED(0.25) value as %69. In our study we obtain better results than above stated results. In Table 2 we obtain a value for MRE as 0, 07461% which is very low for this kind of work and the obtained adjusted R^2 value is very high 99,3%. As a result we believe that we have obtained better results in regression based cost estimation area. Although the estimation function (5) is mathematically well known but it is not used in software cost estimation scope.

7 Discussion and Conclusion

In order to validate the model we adopt an original data set, made of 20 samples from the ISBSG data set which is not used in the training phase of the estimation equation. We obtain results given in Table 2. If we compare the obtained results with the literature they seem to be very impressive. The gap between the results from the study and from the literature is very big. We believe that by using data set from ISBSG and regression models to estimate software project efforts one can obtain results not better than from this study.

We believe that, the use of our estimation model can lead to better software cost estimation and fewer cost overruns in the software industry. In order to increase the realism of the study we have tried to use the real life data from ISBSG data set to make our estimation more realistic for use in real life projects. The next step in our

research will be on understanding the relationship between project characteristics (data set properties) and estimation methods.

Acknowledgements

We would like to thank the International Software Benchmarking Standards Group (ISBSG) to approve for accessing the ISBSG data set for our research.

References

1. Grimstad, S., Jorgensen, M., Østvold, K.M.: Software Effort Estimation Terminology: The tower of Babel. Information and Software Technology 48, 302–310 (2006)
2. Boraso, M., Montangero, C., Sedehi, H.: Software Cost Estimation: an experimental study of model performances, Technical Report: TR-96-22, University of Pisa, Italy
3. Wieczorek, I., Ruhe, M.: How Valuable is company-specific Data Compared to multi-company Data for Software Cost Estimation? In: METRICS 2002. Proceedings of the Eighth IEEE Symposium on Software Metrics (2002)
4. Jones, C.: Applied Software Measurement: Assuring Productivity and Quality. McGraw-Hill, New York (1991)
5. Bontempi, G., Kruijtzer, K.: The use of intelligent data analysis techniques for system-level design: a software estimation example. Soft Computing 8, 477–490 (2004)
6. Jorgensen, M., Shepperd, M.: A Systematic Review of Software Development Cost Estimation Studies. IEEE Transactions On Software Engineering 33(1) (January 2007)
7. Liu, Q., Mintram, R.C.: Preliminary Data Analysis Methods in Software Estimation. Software Quality Journal 13, 91–115 (2005)
8. Stensrud, E., Myrtveit, I.: Human Performance Estimating with Analogy and Regression Models: An Empirical Validation. In: METRICS 1998. Fifth International Symposium on Software Metrics (1998)
9. Hu, Q., Plant, R.T., Hertz, D.B.: Software Cost Estimation Using Economic Production Models. Journal of Management Information System 15(1), 143–163 (1998)
10. Dolado, J.J.: On the problem of the software cost function. Information and Software Technology 43, 61–72 (2001)
11. ISBSG: International Software Benchmarking Standards Group, http://www.isbsg.org
12. Finnie, G.R., Wittig, G.E., Desharnais, J.M.: A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models. Journal of Systems Software 39, 281–289 (1997)
13. Delany, S.J., Cunningham, P., Wilke, N.: The limits of CBR in Software Project Estimation. In: German Workshop on Case-Based Reasoning (1998)
14. Shepperd, M., Schofield, C.: Estimating Software Project Effort Using Anologies. IEEE Transactions on Software Engineering 23(12) (November 1997)

Visualising and Clustering Video Data

Colin Fyfe[1], Wei Chuang Ooi[2], and Hanseok Ko[2]

[1] Applied Computational Intelligence Research Unit,
The University of Paisley, Scotland
colin.fyfe@paisley.ac.uk
[2] Department of Electronics and Computer Engineering,
Korea University, Korea
hsko@korea.ac.kr

Abstract. We review a new form of self-organizing map which is based on a nonlinear projection of latent points into data space, identical to that performed in the Generative Topographic Mapping (GTM) [1]. But whereas the GTM is an extension of a mixture of experts, this model is an extension of a product of experts [6]. We show visualisation and clustering results on a data set composed of video data of lips uttering 5 Korean vowels and show that the new mapping achieves better results than the standard Self-Organizing Map.

1 Introduction

This paper constitutes a part of a much larger study into whether video data which contains both visual and audio information can be used to better transcribe speech data than with audio data alone. Both video and audio data can be very high dimensional - visual data is captured at 20+ frames per second and each frame may contain 10000+ pixels; audio information is generally captured at 8KHz upwards. Both therefore give high dimensional data and we generally wish to process this information in real time. This suggests the need for some form of dimensionality reduction.

We have previously [3,2] investigated linear projections of data sets but such global linear projections may not be able to capture the structure of a data set when the data is either locally low dimensional but globally high dimensional or when the data lies on a nonlinear manifold. We therefore consider nonlinear projections in this paper.

A topographic mapping (or topology preserving mapping) is a transformation which captures some structure in the data so that points which are mapped close to one another share some common feature while points which are mapped far from one another do not share this feature. The most common topographic mappings are Kohonen's self-organizing map (SOM) [10] and varieties of multi-dimensional scaling [5]. The SOM was introduced as a data quantisation method but has found at least as much use as a visualisation tool. It does have the disadvantage that it retains the quantisation element so that while its centres may lie on a manifold, the user must interpolate between the centres to infer the shape of the manifold.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 335–344, 2007.

In this paper, we review two topology preserving mappings, the familiar Self-Organizing Map [10] and the second we call the Topographic Products of Experts (ToPoE)[4]. Based on a generative model of the experts, we show how a topology preserving mapping can be created from a product of experts in a manner very similar to that used by [1] to convert a mixture of experts to the Generative Topographic Mapping (GTM).

We begin with a set of experts who reside in some latent space and take responsibility for generating the data set. In a mixture of experts [8,9], the experts divide up the data space between them, each taking responsibility for a part of the data space. This division of labour enables each expert to concentrate on a specific part of the data set and ignore those regions of the space for which it has no responsibility. The probability associated with any data point is the sum of the probabilities awarded to it by the experts. There are efficient algorithms, notably the Expectation-Maximization algorithm, for finding the parameters associated with mixtures of experts. [1] constrained the experts' positions in latent space and showed that the resulting mapping also had topology preserving properties.

In a product of experts, all the experts take responsibility for all the data: the probability associated with any data point is the (normalised) product of the probabilities given to it by the experts. As pointed out in e.g. [7] this enables each expert to waste probability mass in regions of the data space where there is no data, provided each expert wastes his mass in a different region. The most common situation is to have each expert take responsibility for having information about the data's position in one dimension while having no knowledge about the other dimensions at all, a specific case of which is called a Gaussian pancake in [11]: a probability density function which is very wide in most dimensions but is very narrow (precisely locating the data) in one dimension. It is very elegantly associated with Minor Components Analysis in [11].

In this paper, we review a method of creating a topology preserving mapping from a product of experts, ToPoE. The resulting mapping is neither a true product of experts nor a mixture of experts but lies somewhere in between.

2 SOM

Kohonen's algorithm is exceedingly simple - the network is a simple 2-layer network and competition takes place between the output neurons; however now not only are the prototypes into the winning neuron updated but also the prototypes of its neighbours. Kohonen defined a neighbourhood function $f(i, i^*)$ of the winning neuron i^*. The neighbourhood function is a function of the distance between i and i^*. A typical function is the Difference of Gaussians function; thus if unit i is at point $\mathbf{r}_i$ in the output layer then

$$f(i, i^*) = a \exp(\frac{-|r_i - r_{i^*}|^2}{2\sigma^2}) - b \exp(\frac{-|r_i - r_{i^*}|^2}{2\sigma_1^2})$$

The algorithm is

1. Select at random an input point.
2. There is a competition among the output neurons. That neuron whose protype is closest to the input data point wins the competition:

$$\text{winning neuron, } i^* = \arg\min(\| \mathbf{x} - \mathbf{w_i} \|)$$

3. Now update all neurons' prototypes using

$$\Delta w_{ij} = \alpha(x_j - w_{ij}) * f(i, i^*)$$

4. Go back to the start.

Kohonen typically keeps the learning rate constant for the first 1000 iterations or so and then slowly decreases it to zero over the remainder of the experiment. Two dimensional maps can be created by imagining the output neurons laid out on a rectangular grid or sometimes a hexagonal grid.

3 Topographic Products of Experts

Hinton [6] investigated a product of K experts with

$$p(\mathbf{x}_n|\Theta) \propto \prod_{k=1}^{K} p(\mathbf{x}_n|k) \tag{1}$$

where Θ is the set of current parameters in the model. Hinton notes that using Gaussians alone does not allow us to model e.g. multi-modal distributions, however the Gaussian is ideal for our purposes. Thus our base model is

$$p(\mathbf{x}_n|\Theta) \propto \prod_{k=1}^{K} \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_k - \mathbf{x}_n||^2\right) \tag{2}$$

We will, as with the GTM, allow latent points to have different responsibilities depending on the data point presented:

$$p(\mathbf{x}_n|\Theta) \propto \prod_{k=1}^{K} \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_k - \mathbf{x}_n||^2 r_{kn}\right) \tag{3}$$

where r_{kn} is the responsibility of the k^{th} expert for the data point, $\mathbf{x}_n$. Thus all the experts are acting in concert to create the data points but some will take more responsibility than others. Note how crucial the responsibilities are in this model: if an expert has no responsibility for a particular data point, it is in essence saying that the data point could have a high probability as far as it is concerned. We do not allow a situation to develop where no expert accepts responsibility for a data point; if no expert accepts responsibility for a data

point, they all are given equal responsibility for that data point (see below). For comparison, the probability of a data point under the GTM is

$$p(\mathbf{x}) = \sum_{i=1}^{K} P(i)p(\mathbf{x}|i) = \sum_{i=1}^{K} \frac{1}{K} \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_i - \mathbf{x}||^2\right) \tag{4}$$

We wish to maximise the likelihood of the data set $X = \{\mathbf{x}_n : n = 1, \cdots, N\}$ under this model. The ToPoE learning rule (6) is derived from the minimisation of $-\log(p(\mathbf{x}_n|\Theta))$ with respect to a set of parameters which generate the $\mathbf{m}_k$.

We now turn our attention to the nature of the K experts which are going to generate the K centres, $\mathbf{m}_k$. We envisage that the underlying structure of the experts can be represented by K latent points, $t_1, t_2, \cdots, t_K$. To allow local and non-linear modeling, we map those latent points through a set of M basis functions, $f_1(), f_2(), \cdots, f_M()$. This gives us a matrix Φ where $\phi_{kj} = f_j(t_k)$. Thus each row of Φ is the response of the basis functions to one latent point, or alternatively we may state that each column of Φ is the response of one of the basis functions to the set of latent points. One of the functions, $f_j()$, acts as a bias term and is set to one for every input. Typically the others are gaussians centered in the latent space. The output of these functions are then mapped by a set of weights, W, into data space. W is $M \times D$, where D is the dimensionality of the data space, and is the sole parameter which we change during training. We will use $\mathbf{w}_i$ to represent the i^{th} column of W and Φ_j to represent the row vector of the mapping of the j^{th} latent point. Thus each basis point is mapped to a point in data space, $\mathbf{m}_j = (\Phi_j W)^T$.

We may update W either in batch mode or with online learning. To change W in online learning, we randomly select a data point, say $\mathbf{x}_i$. We calculate the current responsibility of the j^{th} latent point for this data point,

$$r_{ij} = \frac{exp(-\gamma d_{ij}^2)}{\sum_k exp(-\gamma d_{ik}^2)} \tag{5}$$

where $d_{pq} = ||\mathbf{x}_p - \mathbf{m}_q||$, the euclidean distance between the p^{th} data point and the projection of the q^{th} latent point (through the basis functions and then multiplied by W). If no centres are close to the data point (the denominator of (5) is zero), we set $r_{ij} = \frac{1}{K}, \forall j$.

Now we wish to maximise (4) so that the data is most likely under this model. We do this by minimising the -log() of that probability: define $m_d^{(k)} = \sum_{m=1}^{M} w_{md}\phi_{km}$, i.e. $m_d^{(k)}$ is the projection of the k^{th} latent point on the d^{th} dimension in data space. Similarly let $x_d^{(n)}$ be the d^{th} coordinate of $\mathbf{x}_n$. These are used in the update rule

$$\Delta_n w_{md} = \sum_{k=1}^{K} \eta\phi_{km}(x_d^{(n)} - m_d^{(k)})r_{kn} \tag{6}$$

where we have used Δ_n to signify the change due to the presentation of the n^{th} data point, $\mathbf{x}_n$, so that we are summing the changes due to each latent point's

response to the data points. Note that, for the basic model, we do not change the Φ matrix during training at all.

3.1 Comparison with the GTM

The Generative Topographic Mapping (GTM) [1] is a mixture of experts model which treats the data as having been generated by a set of latent points. These K latent points are also mapped through a set of M basis functions and a set of adjustable weights to the data space. The parameters of the combined mapping are adjusted to make the data as likely as possible under this mapping. The GTM is a probabilistic formulation so that if we define $\mathbf{y} = \mathbf{\Phi W} = \mathbf{\Phi}(\mathbf{t})\mathbf{W}$, where $\mathbf{t}$ is the vector of latent points, the probability of the data is determined by the position of the projections of the latent points in data space and so we must adjust this position to increase the likelihood of the data. More formally, let

$$\mathbf{m}_i = \mathbf{\Phi}(\mathbf{t}_i)W \tag{7}$$

be the projections of the latent points into the feature space. Then, if we assume that each of the latent points has equal probability

$$p(\mathbf{x}) = \sum_{i=1}^{K} P(i)p(\mathbf{x}|i) = \sum_{i=1}^{K} \frac{1}{K}\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_i - \mathbf{x}||^2\right) \tag{8}$$

where D is the dimensionality of the data space. i.e. all the data is assumed to be noisy versions of the mapping of the latent points. This equation should be compared with (3) and (4).

In the GTM, the parameters W and β are updated using the EM algorithm though the authors do state that they could use gradient ascent. Indeed, in the ToPoE, the calculation of the responsibilities may be thought of as being a partial E-step while the weight update rule is a partial M-step. The GTM has been described as a "principled alternative to the SOM" however it may be criticised on two related issues:

1. it is optimising the parameters with respect to each latent point independently. Clearly the latent points interact.
2. using this criterion and optimising the parameters with respect to each latent point individually does not necessarily give us a globally optimal mapping from the latent space to the data space.

The ToPoE will overcome the second of these shortcomings in that all data points are acting together. Specifically if no latent point accepts responsibility for a data point, the responsibility is shared equally amongst all the latent points.

The GTM, however, does have the advantage that it can optimise with respect to β as well as W. However note that, in (3) and (4), the variance of each expert is dependent on its distance from the current data point via the hyper-parameter, γ. Thus we may define

$$(\beta_k)_{|\mathbf{x}=\mathbf{x}_n} = \beta r_{kn} = \beta \frac{exp(-\gamma d_{nk}^2)}{\sum_t exp(-\gamma d_{nt}^2)} \tag{9}$$

Therefore the responsibilities are adapting the width of each expert locally dependent on both the expert's current projection into data space and the data point for which responsibility must be taken. Initially, $r_{kn} = \frac{1}{K}, \forall k, n$ and so we have the standard product of experts. However during training, the responsibilities are refined so that individual latent points take more responsibility for specific data points. We may view this as the model softening from a true product of experts to something between that and a mixture of experts.

A model based on products of experts has some advantages and disadvantages. The major disadvantage is that no efficient EM algorithm exists for optimising parameters. [6] suggests using Gibbs sampling but even with the very creative method discussed in that paper, the simulation times were excessive. Thus we have opted for gradient descent as the parameter optimisation method.

The major advantage which a product of experts method has is that it is possible to get very much sharper probability density functions with a product rather than a sum of experts.

4 Visualising and Clustering Voice Data

This work is part of a larger body of work in which we wish to combine audio and video data in order to better transcribe speakers audio utterances. As part of this work, we investigated clustering and visualisation of the video data alone.

4.1 The Data and Pre-processing

14 speakers were asked to utter each of 5 Korean vowels and were videoed while doing so. The five vowels were approximately

1. 'ah' as in the English word, 'cat'
2. 'eh' as in the English word, 'bed'
3. 'ee' as in the English word, 'feel'
4. 'oh' as in the English word, 'door'
5. 'wu'[1] as in the English word, 'wood'

Each speaker spoke for approximately 1 second on each vowel and between 21 and 27 frames were taken. The video sequences were cropped to a 71× 71 pixel region round the mouth so that we have 5041 dimensional data, each pixel of which is in a range from 0 to 255.

The lighting conditions were very different from speaker to speaker and so we first normalised each video so that the pixels varied from -1 to 1 (this is a very crude way to perform this but we wished to minimise the pre-processing requirements). We then performed a principal component analysis of the data and, based on the variances, opted to investigate further processing based on the projection of the data onto 4 and 10 principal components. In practise, there was little difference in the results and in this paper we use the first 10 principal components. Thus we have compressed our 5041 dimensional data down to 10 dimensions and it is in this data that we look for structure.

[1] The Korean language *does* have an initial 'w' associated with this sound.

4.2 Experiments

We first use each frame as a separate sample: in Figure 1, we show the projections of the data found by ToPoE. We see that there is some structure in the mapping - the top half is dominated by the open lip data ('ah','eh' and 'ee') and the bottom half is dominated by the closed lip data ('oh' and 'wu'). However there is a great deal of overlap between these. This is caused by the fact that in all videos the subjects began the vocalisation in a similar pose. Also a nearest neighbour investigation in this space showed that often the nearest neighbour was a frame of the same person but speaking a different vowel. We therefore subsequently selected the first 21 frames of each of the videos and concatenated these to form one data sample of dimensionality 210 (21 frames of the 10 principal components). Note that this is not the same as performing a principal component analysis of the completed data set (which would have involved a PCA of 21*5041 dimensional data) but is an attempt to capture some essential features of the data.

Therefore we now have 70 samples (14 speakers each saying 5 vowels) of 210 dimensional data. The SOM projection of this data is shown in Figure 2: we see a very good separation of the open mouth vowels from the rounded mouth vowels but it is not perfect - there is some overlap between the two groups.

We can alleviate this by using the audio data too. Each audio signal consisted of between 10000 and 16000 samples. We therefore select the first 10000 samples of the audio signal of each video and concatenate these to create a 10000×70 data set. We again performed a PCA on this data set and projected each sample onto the first 10 principal components.

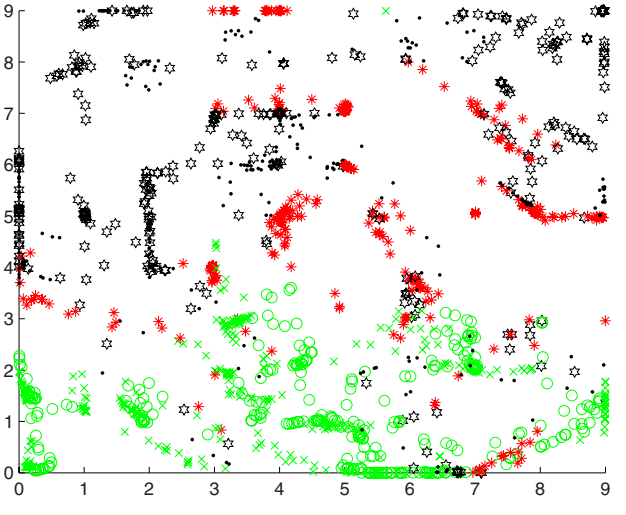


Fig. 1. The ToPoE projection of the visual projections of the lips data. The black stars are 'ah', the red asterisks are 'eh', black dots 'ee', green circles 'oh' and green crosses 'wu'.

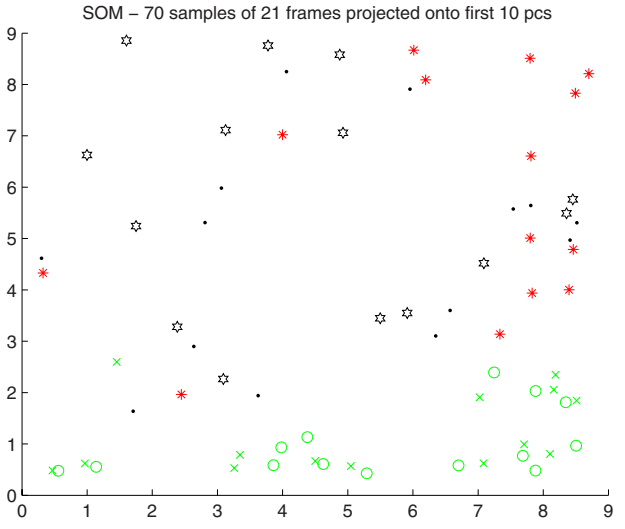


Fig. 2. The SOM projection of the video data when we use 21 frames of the first 10 principal components as 1 data sample. We now have 70 samples = 14 speakers of 5 vowels.

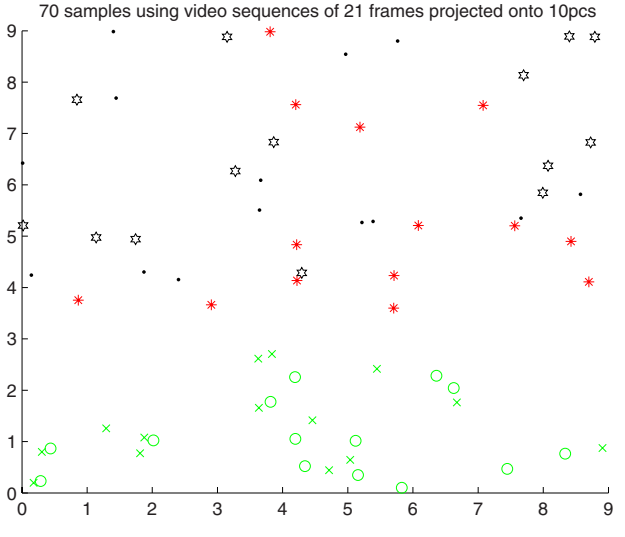


Fig. 3. The SOM projection of the combined audio and visual data. A clearer separation of the two groups of vowels is achieved.

Figure 3 shows the SOM projection when we use 70 samples (14 speakers of 5 vowels) with the combined audio and video data. We see a far better separation of the two groups of vowels; note that treating this data as two separate data

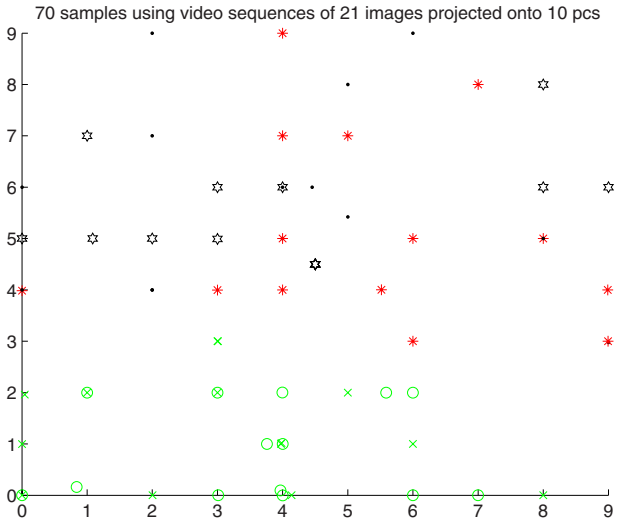


Fig. 4. The ToPoE clearly separates the two groups of vowels very clearly

streams which can be subsequently conjoined means that we do not have to worry about the problem of matching the audio and visual data streams in time. However this process left us somewhat dissatisfied in that our original investigation was into utilising the information from the visual data to assist the transcription of the audio data. The results here certainly show that we can use one to assist in differentiating the other but we are actually using the audio data to assist in optimising the projection of the visual data.

We therefore investigate the use of the ToPoE on only the visual data as above. The results are shown in Figure 4: the two groups of vowels are clearly separated using only the visual data.

5 Conclusion

We investigated the task of finding a good projection of visual data so that different classes of visual data can be clearly identified. We have shown that the Topographic Product of Experts gives a better projection than the standard Self-Organizing Map, though if we add audio information the difference between the mappings is much less.

Future work will continue on these lines with a larger data set, the current work being only a proof of concept. We will also investigate other projections of these data sets such as principal curves, isomap and so on.

Acknowledgement. This research was supported by the MIC (Ministry of Information and Communication), Korea, Under the ITFSIP (IT Foreign Specialist

Inviting Program) supervised by the IITA (Institute of Information Technology Advancement).

References

1. Bishop, C.M., Svensen, M., Williams, C.K.I.: Gtm: The generative topographic mapping. Neural Computation (1997)
2. Corchado, E., MacDonald, D., Fyfe, C.: Maximum and minimum likelihood hebbian learning for exploratory projection pursuit. Data Mining and Knowledge Discovery 8, 203–225 (2004)
3. Fyfe, C.: A comparative study of two neural methods of exploratory projection pursuit. Neural Networks 10(2), 257–262 (1997)
4. Fyfe, C.: Two topographic maps for data visualization. Data Mining and Knowledge Discovery 14, 207–224 (2007)
5. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer, Heidelberg (2001)
6. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Technical Report GCNU TR 2000-004, Gatsby Computational Neuroscience Unit, University College, London (2000), http://www.gatsby.ucl.ac.uk/
7. Hinton, G.E., Teh, Y.-W.: Discovering multiple constraints that are frequently approximately satisfied. In: Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence, pp. 227–234 (2001)
8. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Computation 3, 79–87 (1991)
9. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the em algorithm. Neural Computation 6, 181–214 (1994)
10. Kohonen, T.: Self-Organising Maps. Springer, Heidelberg (1995)
11. Williams, C., Agakov, F.V.: Products of gaussians and probabilistic minor components analysis. Technical Report EDI-INF-RR-0043, University of Edinburgh (2001)

Neural Network-Based Receiver for Uplink Multiuser Code Division Multiple Access Communication System

Zi-Wei Zheng

College of Information Engineering, Dalian Maritime University, 116026 Dalian, P.R. China
ziwei_zheng@yahoo.com.cn
Nation Mobile Communications Research Laboratory, Southeast University, 210096 Nanjing, P.R. China

Abstract. In this paper, the uplink multiuser code division multiple access (CDMA) communication system model is described in the form of space-time domain through antenna array and multipath fading expression. Novel suitable neural network technique is proposed as an effective signal processing method for the receiver of such an uplink multiuser CDMA system. By the appropriate choice of the channel state information for the neural network parameters, the neural network can collectively resolve the effects of both the inter-symbol interference due to the multipath fading channel and the multiple access interference in the receiver of the uplink multiuser CDMA communication system. The dynamics of the proposed neural network receiver for the uplink multiuser CDMA communication system is also studied.

Keywords: neural network, receiver, uplink multiuser CDMA system.

1 Introduction

Wireless communications for mobile telephone and data transmission is currently undergoing very rapid development. Many of the emerging wireless systems will incorporate considerable signal processing intelligence in order to provide advanced services such as multimedia transmission [1]. In order to make optimal use of available bandwidth and to provide maximal flexibility, many wireless systems operate as multiple-access systems, in which channel bandwidth is shared by many users on a random access basis. One type of multiple access technique that has become very popular in recent years is code division multiple access (CDMA). CDMA implemented with direct sequence spread spectrum signaling is among the most promising multiplexing technologies for cellular telecommunications services, such as personal communications, mobile telephony, and indoor wireless networks. The advantages of direct-sequence spread-spectrum for these services include superior operation in multipath fading environments, flexibility in the allocation of channels, the ability to operate asynchronously, increased capacity in bursty or fading channels, and the ability to share bandwidth with narrow-band communication systems without undue degradation of either system's performance [2]. CDMA or wideband CDMA is one of the more promising candidates for third-generation (3G)

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 345–355, 2007.

or beyond 3G cellular services [3-4]. Inter-symbol interference (ISI) and multiple access interference (MAI) are two major problems in CDMA or wideband CDMA. Different advanced signal processing techniques have been proposed to deal with these problems. Multiuser detection [5] and space-time processing [6] are the two main category techniques to combat interference and multipath fading channel distortion.

A variety of multiuser detectors have been proposed to deal with MAI through demodulation of mutually interfering signals. However, the computational effort required for the solution of optimum receiver by using conventional multiuser detection and space-time processing methods becomes prohibitively large for real-time implementations [2], [5].

Neural networks [7], [8] are layer networks with output feedback consisting of simple processors (neurons) that can collectively provide good solutions to difficult optimization problems.

With the advent of neural networks, detectors based on well-known structures like multilayer perceptrons [9] were proposed. With increasing attention focused on the application of neural networks to the field of pattern recognition, more neural network–based multiuser detectors were implemented. In [10], [11], Miyajima and Kechriotis uses neural network as the signal processing method for the multiuser receiver under AWGN channel environment and shows to have excellent sub-optimum performance. However, the neural network signal processing method proposed in [10], [11] is unable to exhibit good performance for the uplink multiuser CDMA communication system with both multiple access interference (MAI) and inter-symbol interference (ISI) due to the multipath fading channel. The reason behind this is that neural network parameters set in [10], [11] are not considering the whole channel characteristics of the uplink multiuser CDMA communication system. Recently, blind detectors [12] and kernel-based detectors [13] have been investigated.

In this paper, we consider the signal processing problem by using neural network technique for the receiver of the uplink multiuser CDMA system with both multiple access interference (MAI) and inter-symbol interference (ISI) due to the multipath fading channel. The uplink multiuser CDMA communication system model is described in the form of space-time domain through antenna array and multipath fading expression. By the appropriate choice of the channel state information for the neural network parameters, the neural network can collectively resolve the effects of both the multipath fading and the multiple access interference for the receiver of the uplink multiuser CDMA system.

The rest of the paper is organized as follows. In Section 2, the model of uplink multiuser CDMA communication system is presented. In Section 3, neural network technique is proposed as an effective signal processing method for the receiver of the uplink multiuser CDMA communication system. In Section 4, simulation studies are performed for the evaluation of the system performance of the uplink multiuser CDMA communication systems under different channel situations. In Section 5, conclusions are given.

In what follows, boldface capital (lowercase) letters refer to matrices (vectors), the superscript $(\cdot)^T$ denotes the transpose operation.

2 Communication System Model

The uplink in an asynchronous direct sequence code division multiple access (DS-CDMA) cellular mobile radio network with K active users is considered.

The transmitted baseband signal due to the kth user is given by

$$x_k(t) = A_k \sum_{i=0}^{M-1} b_k[i] s_k(t - iT - q_k),$$

$$k = 1,2,\cdots,K \tag{1}$$

where M, T, $b_k[i]$, respectively denotes, the number of data symbols per frame, the symbol interval, the ith transmitted symbol by the kth user. It is assumed that for each user k, the symbol stream $\{b_k[i]\}$ is a collection of independent equiprobable ± 1 random variables, and the symbol streams of different users are independent. A_k, $s_k(t)$, q_k $(0 \le q_k < T)$, respectively denotes, the amplitude, the normalized signaling waveform, the delay of the kth user's signal.

At the base station receiver, a uniform linear antenna array of P elements is employed. Assume that the channel can be modeled as a tapped delay line with L complex coefficients as the number of resolvable multipaths. The baseband multipath channel between the kth user's transmitter and the base station receiver can be modeled as a single-input multiple-output channel in the form of space-time domain through antenna array and multipath expression.

When the fractionally sampled (oversampled) received signals are used in the digital receivers of the mobile wireless communication systems, oversampling gives rise to cyclostationarity (CS) and provides more statistical information which can be used to improve the communication system performance [2]. Therefore, chip oversampling is used in the base station receiver of the uplink CDMA mobile communication systems. The total received signal vector $\mathbf{y}(t)$, superposed all the users' signals and additive Gaussian noise, is sampled at a multiple ($\overline{m}$) of the chip-rate, i.e., the sampling time interval is $\Delta = (T_c/\overline{m}) = (T/\overline{M})$, where $\overline{M} = \overline{m}N$ is the total number of samples per symbol interval, $\overline{m}$ is the chip oversampling factor.

Denote

$$\mathbf{g}_k(t) = \sum_{j=0}^{N-1} A_k \sum_{l=1}^{L} \mathbf{a}_{kl} g_{kl} \psi(t - jT_c - q_k - \tau_{kl}) \tag{2}$$

$$\mathbf{h}_k(t) = \sum_{j=0}^{N-1} c_k[j] \mathbf{g}_k(t) \tag{3}$$

$$\mathbf{n}(t) = [n_1(t), n_2(t), \cdots, n_P(t)]^T \tag{4}$$

$$\nu_k = \left\lceil \frac{q_k + \tau_{kl} + T_c}{T} \cdot \frac{T}{T_c} \right\rceil \le \bar{t}_k N \tag{5}$$

$$\bar{t}_k = \lceil (q_k + \tau_{kl} + T_c)/T \rceil \tag{6}$$

$$\bar{t} = \max_{1\le k\le K}\{\bar{t}_k\} \tag{7}$$

$$m_{smoothing} = \left\lceil (\overline{M}+K)/(\overline{M}-K)\right\rceil \bar{t} \tag{8}$$

where $\mathbf{g}_k(t)$ is the composite channel response vector, taking into account the effects of transmitted power, antenna array response, chip pulse waveform, and the multipath channels. N, $\{c_k[n]\}_{n=0}^{N-1}$, $\psi(t)$, respectively denotes, the processing gain, the binary (± 1) spreading code, the normalized chip waveform of duration T_c. $\mathbf{a}_{kl} = [a_{kl,1}, a_{kl,2}, \cdots, a_{kl,P}]^T$, g_{kl}, τ_{kl}, respectively denotes, the array response vector, the complex gain, the delay, corresponding to the lth path of the kth user's signal. $n_p(t)$ $(1 \le p \le P)$ is the additive Gaussian noise at the pth antenna. $m_{smoothing}$ is the smoothing factor.

Denote following discrete vectors and matrices

$$\mathbf{h}_{k,j} = \left[\mathbf{h}_k[j,0], \mathbf{h}_k[j,1], \cdots, \mathbf{h}_k[j,\overline{M}-1]\right]^T \tag{9}$$

$$\mathbf{g}_{k,j} = \left[\mathbf{g}_k[j,0], \mathbf{g}_k[j,1], \cdots, \mathbf{g}_k[j,\overline{M}-1]\right]^T \tag{10}$$

$$\mathbf{h}_k[j,n] = \sum_{\bar{i}=0}^{N-1} c_{k,j}[\bar{i}]\,\mathbf{g}_k(jT + n\Delta - lT_c) \tag{11}$$

$$\mathbf{g}_k[j,n] = \mathbf{g}_k(jT + n\Delta - lT_c) \tag{12}$$

$$\mathbf{h}_{k,j} = \mathbf{C}_{k,j}\mathbf{g}_{k,j} \tag{13}$$

$$\mathbf{C}_{k,j} = \mathrm{diag}\{\overline{\mathbf{C}}_{k,j}[0], \overline{\mathbf{C}}_{k,j}[1], \cdots, \overline{\mathbf{C}}_{k,j}[\overline{M}-1]\} \tag{14}$$

$$\overline{\mathbf{C}}_{k,j}[n] = \begin{bmatrix} c_{k,j,1}[n] & 0 & \cdots & 0 \\ 0 & c_{k,j,2}[n] & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & c_{k,j,P}[n] \end{bmatrix} \tag{15}$$

$$\mathbf{H}_j = [\mathbf{h}_{1,j}, \mathbf{h}_{2,j}, \cdots, \mathbf{h}_{K,j}] \tag{16}$$

$$\mathbf{y}_j[i,n] = \sum_{k=1}^{K} \mathbf{h}_k[j,n] b_k[i-j] + \sigma\mathbf{n}_j[i,n] \tag{17}$$

$$\mathbf{b}_j[i] = \begin{bmatrix} b_1[i-j] \\ \vdots \\ b_K[i-j] \end{bmatrix} \tag{18}$$

$$\overline{\mathbf{y}}_j[i]=\begin{bmatrix}\mathbf{y}_j[i,0]\\ \vdots \\ \mathbf{y}_j[i,\overline{M}-1]\end{bmatrix} \tag{19}$$

$$\overline{\mathbf{n}}_j[i]=\begin{bmatrix}\mathbf{n}_j[i,0]\\ \vdots \\ \mathbf{n}_j[i,\overline{M}-1]\end{bmatrix} \tag{20}$$

$$\overline{\mathbf{y}}[i]=\begin{bmatrix}\overline{\mathbf{y}}_0[i]\\ \vdots \\ \overline{\mathbf{y}}_{\bar{t}_k}[i]\end{bmatrix} \tag{21}$$

$$\overline{\mathbf{n}}[i]=\begin{bmatrix}\overline{\mathbf{n}}_0[i]\\ \vdots \\ \overline{\mathbf{n}}_{\bar{t}_k}[i]\end{bmatrix} \tag{22}$$

$$\mathbf{b}[i]=\begin{bmatrix}\mathbf{b}_0[i]\\ \vdots \\ \mathbf{b}_{\bar{t}_k}[i]\end{bmatrix} \tag{23}$$

$$\overline{\mathbf{H}}=\begin{bmatrix}\mathbf{H}_0 & \cdots & \mathbf{H}_{\bar{t}_k} & \cdots & \mathbf{0}\\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{H}_0 & \cdots & \mathbf{H}_{\bar{t}_k}\end{bmatrix} \tag{24}$$

where $j=0,1,\cdots,\bar{t}_k$, $n=0,1,\cdots,\overline{M}-1$, diag$\{\ \}$ denotes diagonal operation, $\overline{\mathbf{C}}_{k,j}[n]$ is a $P\times P$ matrix, $\overline{\mathbf{H}}$ is a $P\overline{m}N(\bar{t}_k+1)\times K(\bar{t}_k+1)$ matrix. In this paper, we let the chip oversampling factor $\overline{m}=2$, then, we have a $2PN\times 2PN$ matrix $\mathbf{C}_{k,j}=\text{diag}\{\overline{\mathbf{C}}_{k,j}[0],\overline{\mathbf{C}}_{k,j}[1],\cdots,\overline{\mathbf{C}}_{k,j}[2N-1]\}$. For other values of $\overline{m}$, the matrix $\mathbf{C}_{k,j}$ can be similarly constructed. Suppose that user k is the user of interest, and at every based station received antenna p, ($p=1,2,\cdots,P$), for every propagation path j, ($j=0,1,\cdots,\bar{t}_k$), the spreading sequence $\{c_{k,j,p}[0],c_{k,j,p}[1],\cdots,c_{k,j,p}[2N-1]\}$ is known to the receiver (and therefore $\mathbf{C}_{k,j}$ is known to the receiver). Then, the received discrete signals can be expressed in a matrix form as

$$\overline{\mathbf{y}}[i]=\overline{\mathbf{H}}\,\mathbf{b}[i]+\sigma\overline{\mathbf{n}}[i] \tag{25}$$

where $\overline{\mathbf{y}}[i]$, $\overline{\mathbf{n}}[i]$, $\mathbf{b}[i]$, $\overline{\mathbf{H}}$ are, respectively, a $P\overline{m}N(\bar{t}_k+1)\times 1$ received signal vector, a $P\overline{m}N(\bar{t}_k+1)\times 1$ independent zero-mean complex white Gaussian noise vector with variance of σ^2, a $K(\bar{t}_k+1)\times 1$ transmitted symbol vector, a $P\overline{m}N(\bar{t}_k+1)\times K(\bar{t}_k+1)$ multipath channel matrix.

3 Neural Network-Based Receiver

From equation (25), based on the minimization of the likelihood function, the transmitted symbol vector of the optimal receiver can be estimated as [5]

$$\hat{\mathbf{b}}[i] = \arg\min\left\{\left\|\bar{\mathbf{y}}[i] - \overline{\mathbf{H}}\,\mathbf{b}[i]\right\|^2\right\} = \arg\min\left\{-\bar{\mathbf{y}}[i]^T\overline{\mathbf{H}}\mathbf{b}[i] + \frac{1}{2}\mathbf{b}[i]^T\overline{\mathbf{H}}^T\overline{\mathbf{H}}\,\mathbf{b}[i]\right\} \tag{26}$$

Note that if the value of $P\overline{m}N(\bar{t}_k + 1)$ is relatively large, even for a small-to-moderate number of users K, the computational effort required for the solution of (26) becomes prohibitively large for real-time implementations.

Neural networks are layer networks with output feedback consisting of simple processors (neurons) that can collectively provide good solutions to difficult optimization problems. Neural networks have been employed extensively to solve a variety of difficult combinatorial optimization problems [7]-[19].

Next, we will transform the minimization of the likelihood function given in (26) into the minimization of neural network energy function E_{NN} described by the expression

$$E_{\mathrm{NN}} = -\mathbf{OI} - \frac{1}{2}\mathbf{OMO} \tag{27}$$

By setting $\mathbf{I} = \bar{\mathbf{y}}[i]\overline{\mathbf{H}}^T$ and $\mathbf{M} = -\overline{\mathbf{H}}^T\overline{\mathbf{H}}$, then in (27), $\mathbf{O}$ is the output of neural network neurons and $\mathbf{M}$ is the interconnection matrix between neural network neurons.

Once the above transformation is done, the sub-optimum estimation of the transmitted symbol vector could be simply driven from the neural network receiver output by using

$$\hat{\mathbf{b}}[i] \approx \mathbf{O} \tag{28}$$

From equation (25) to equation (28), we can see that by the appropriate choice of the channel state information used for the neural network parameters, the neural network can collectively resolve the effects of both the inter-symbol interference due to the multipath fading and the multiple access interference in the receiver of the uplink multiuser CDMA communication system.

The channel state information is very crucial to the proposed neural network receiver of the uplink multiuser CDMA communication system. Imperfections in channel state information degrade the neural network receiver performance. Channel estimation can be achieved by sending training sequences, using pilot channel, or using blind methods. Periodic transmission of training sequences make the identification of channel state information feasible since both input and output signals are known during the transmission of these sequences.

In the following, more information about the dynamics of neural network receiver will be discussed.

The dynamic equation implemented by the neural network energy function E_{NN} is

$$C_i \frac{do_i}{dt} = -\frac{\partial E_{\mathrm{NN}}}{\partial o_i}(\mathbf{O}) - G_i o_i \ , \ i = 1, \cdots, P\overline{m}N(\bar{t}_k + 1) \tag{29}$$

Where C_i is the output capacity and G_i is the parasite conductance of neuron i. Assuming for simplicity that $C_i = C$ and $G_i = G$ for all the neurons, then, the dynamic behavior of the neural network energy function E_{NN} is

$$\frac{dE_{NN}}{dt} = -\sum_{i=1}^{P\overline{m}N(\bar{t}_k+1)} C\left(\frac{do_i}{dt}\right)^2 \tag{30}$$

Equation (30) shows that the neural network energy function E_{NN} goes toward the minimum value. Thus, for any initial value, the neural network evolves toward the minimum, and the energy function has a global minimum point.

The gradient of the neural network energy function E_{NN} is

$$\nabla E_{NN}(\mathbf{O}) = -\mathbf{I} - \mathbf{M}\mathbf{O} \tag{31}$$

Equation (31) shows that the neural network energy function E_{NN} has a unique minimum value $\hat{\mathbf{O}}_{\mathrm{opt}} = -\mathbf{M}^{-1}\mathbf{I}$. It is important to note that the minimum value derived according to the above procedure equals to the optimum minimum value

$$\mathbf{O}_{\mathrm{opt}} = -\mathbf{M}^{-1}\mathbf{I} \tag{32}$$

From above discussion, we can see that the neural network energy function has a global minimum point and is stable for any initial value.

Therefore, the implementation of the neural network based receiver proposed for the uplink multiuser CDMA communication system is feasible.

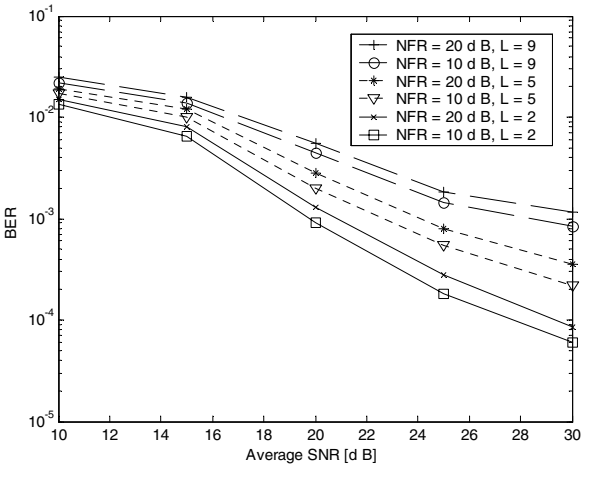


Fig. 1. System performance of matched-filter-based neural network method proposed in [10], [11] under different channel situations

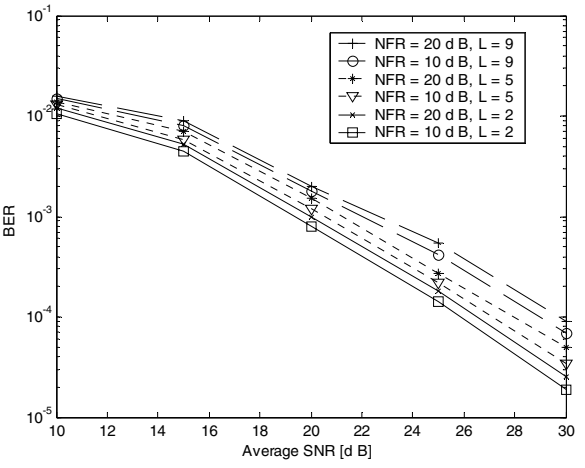


Fig. 2. System performance of channel-state-information-parameters-assisted neural network method proposed in this paper under different channel situations

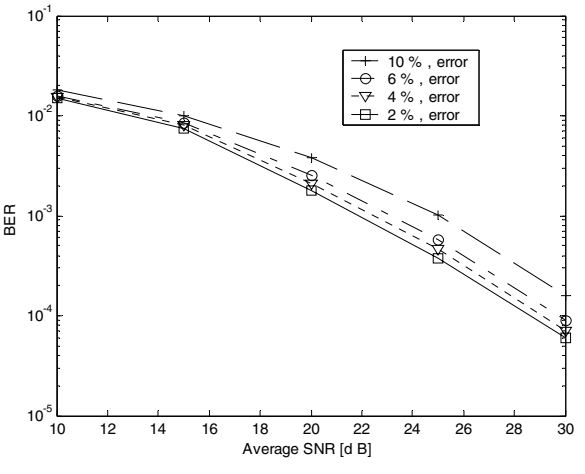


Fig. 3. System performance of channel-state-information-parameters-assisted neural network method proposed in this paper with different errors of the channel state information parameters

4 Performance Evaluation

The simulated CDMA system is an asynchronous system adopting $N = 31$ Gold codes as spreading sequences, with users $K = 15$, antenna elements $P = 3$, multipath diversity order $L = 2$, $L = 5$ or $L = 9$, the number of symbols per frame is $M = 300$, the chip pulse is a raised cosine pulse with roll-off factor 0.5, the initial delay q_k of

each user is uniformly generated on $[0, LT_c]$, the delay of each path τ_{kl} is uniformly generated on $[0, LT_c]$. The modulation scheme is BPSK.

In the following, the system performance studies of the method proposed in [10], [11] (we call it the matched-filter-based neural network method) are provided for the references with the proposed method in this paper (we call it the channel-state-information-parameters-assisted neural network method).

When the matched-filter-based neural network method proposed in [10], [11] is used as the signal processing method, the simulated bit error rate (BER) versus average signal-to-noise ratio (SNR) performance is shown in figure 1 for different near-far ratio (NFR) and different multipath diversity order L.

From figure 1, we can see that the BER versus average SNR performance of the matched-filter-based neural network method proposed in [10], [11] is unable to resistant to the inter-symbol interference (ISI) due to the multipath fading channel (with different multipath diversity order L). When the average SNR is larger than 20 dB, the BER versus average SNR performance of the matched-filter-based neural network method proposed in [10], [11] becomes much worse as the multipath diversity order is increasing.

When the channel-state-information-parameters-assisted neural network method proposed in this paper is used as the signal processing method, the simulated bit error rate (BER) versus average signal-to-noise ratio (SNR) performance is shown in figure 2 for different near-far ratio (NFR) and different multipath diversity order L. (The channel estimation method proposed in [20] is used here to estimate the channel state information for the neural network parameters.)

From figure 2, we can see that the channel-state-information-parameters-assisted neural network method proposed in this paper can collectively resolve the effects of both the multipath fading (with different multipath diversity order L) and the multiple access interference (with different near-far ratio NFR). When the average SNR is larger than 20 dB, the BER versus average SNR performance of the channel-state-information-parameters-assisted neural network method proposed in this paper becomes only a little worse as the multipath diversity order is increasing and the near-far ratio is increasing.

Since the accuracy of the channel state information for the neural network parameters is very crucial to the channel-state-information-parameters-assisted neural network method proposed in this paper, the system performance affected by the error in the channel state information parameters will be studied in the following.

When near-far ratio (NFR) is $\text{NFR} = 10$ dB and multipath diversity order is $L = 5$, the average simulated bit error rate (BER) versus signal-to-noise ratio (SNR) system performance for the case in which the channel state information parameters delivered to the proposed channel-state-information-parameters-assisted neural network receiver contains different errors are shown in figure 3.

From figure 3, we can see that the less the error in the channel state information parameters, the better the channel-state-information-parameters-assisted neural network receiver performance.

5 Conclusions

In this paper, a novel suitable neural network technique is proposed as an effective signal processing method for the receiver of the uplink multiuser CDMA system. The dynamics of the proposed neural network receiver for the uplink multiuser CDMA communication system is discussed. Simulation studies show that the proposed neural network receiver can collectively resolve the effects of both the inter-symbol interference due to the multipath fading channel and the multiple access interference in the receiver of the uplink multiuser CDMA communication system if the channel state information for the neural network parameters is appropriately chosen. The accuracy of the channel state information for the neural network parameters is crucial to the system performance of the proposed neural network receiver for the uplink multiuser CDMA communication system.

Acknowledgments

This work was supported in part by the National Science Foundation of China (No. 60602021, No. 60772119), the Specialized Research Fund for the Doctoral Program of Higher Education (No. 20060151001), the Excellence Young Talent Fund of Dalian (No. 2006J23JH037), the open research fund of Nation Mobile Communications Research Laboratory of Southeast University (No. A200606).

References

1. Dimitriou, N., Tafazolli, R., Sfikas, G.: Quality of service for multimedia CDMA. IEEE Commun. Mag. 38(7), 88–94 (2000)
2. Wang, X., Poor, H.V.: Wireless communication systems: advanced techniques for signal reception. Prentice Hall, Upper Saddle River, NJ (2004)
3. Zheng, Z.W., Yang, Z.X., Zhu, Y.S., Pan, C.Y.: Channel estimation and interference suppression for uplink CDMA mobile communication systems. Wireless Communications and Mobile Computing 4(5), 483–489 (2004)
4. Ojanperä, T., Prasad, R.: Wideband CDMA for Third Generation Mobile Communications. Artech House, Norwood, MA (1998)
5. Verdu, S.: Multiuser Detection. Cambridge University Press, UK (1998)
6. Paulraj, A.J., Williams, C.B.: Space-time processing for wireless communications. IEEE Signal Process. Mag. 14(6), 49–83 (1997)
7. Hopfield, J.J.: Neural networks and physical systems with emerging collective computational abilities. Proc. Nut. Acad. Sci. 79, 2554–2558 (1982)
8. Hopfield, J.J., Tank, D.W.: Neural computation of decisions in optimization problems. Biol. Cybern. 52, 141–152 (1985)
9. Shayesteh, M.G., Amindavar, H.: Multiuser detection in DS/CDMA systems using neural networks. In: IEEE 7th Int. Symp. Spread-Spectrum Tech. Appl., pp. 506–510 (September 2002)
10. Miyajima, T., Hasegawa, T.: Multiuser detection using a Hopfield network for asynchronous code-division multiple-access systems. IEICE Trans. Fund. Elect. Commun. Comp. Sci. E79-A(12), 1963–1971 (1996)

11. Kechriotis, G.I., Manolakos, E.S.: Hopfield neural network implementation of the optimal CDMA multiuser detector. IEEE Trans. Neural Networks 7(1), 131–141 (1996)
12. Waheed, K., Salem, F.M.: Blind information theoretic multiuser detection algorithms for DS-CDMA and WCDMA downlink systems. IEEE Trans. Neural Netw. 16(4), 937–948 (2005)
13. Chen, S., Hanzo, L.: Block-adaptive kernel-based CDMA multiuser detection. In: Proc. IEEE Int. Conf. Commun., pp. 682–686 (2002)
14. Smith, K., Palaniswami, M., Krishnamoorthy, M.: Neural techniques for combinatorial optimization with applications. IEEE Trans. Neural Networks 9(6), 1301–1318 (1998)
15. Zhu, W., Liang, T.Y., Shieh, C.K.: A Hopfield neural network based task mapping method. Computer Commun. 22, 1068–1079 (1999)
16. Engelhart, A., Teich, W.G., Lindner, J., Jeney, G., Imre, S., Pap, L.: A survey of multiuser/multisubchannel detection schemes based on recurrent neural networks. Wireless Commun Mobile Comput. 2, 269–284 (2002)
17. Sanz, S.S., Mozos, R.S., Calzón, C.B.: A hybrid Hopfield network-simulated annealing approach for frequency assignment in satellite communications systems. IEEE Trans. Syst., Man, Cybern. B 34, 1108–1116 (2004)
18. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall, Upper Saddle River, N.J. (1999)
19. Haykin, S.: Adaptive Filter Theory, 3rd edn. Prentice-Hall, Englewood Cliffs, NJ. (1996)
20. D'Amico, A.A., Mengali, U., Morelli, M.: Channel estimation for the uplink of a DS-CDMA system. IEEE Trans. Wireless Commun. 2(6), 1132–1137 (2003)

Evolving Tree Algorithm Modifications

Vincenzo Cannella[1], Riccardo Rizzo[2], and Roberto Pirrone[1]

[1] DINFO - University of Palermo
Viale delle Scienze, 90128 Palermo, Italy
[2] ICAR - Italian National Research Council
Viale delle Scienze, 90128 Palermo, Italy
ricrizzo@pa.icar.cnr.it

Abstract. There are many variants of the original self-organizing neural map algorithm proposed by Kohonen. One of the most recent is the Evolving Tree, a tree-shaped self-organizing network which has many interesting characteristics. This network builds a tree structure splitting the input dataset during learning. This paper presents a speed-up modification of the original training algorithm useful when the Evolving Tree network is used with complex data as images or video. After a measurement of the effectiveness an application of the modified algorithm in image segmentation is presented.

1 Introduction

Growing neural networks are capable to adapt the number of neural units to the input patterns distribution. Many growing neural nets add new units at fixed pace during the learning procedure: new units are added near the unit that accumulates the greater approximation error (as in Growing Neural Gas [8]) or when there is not a neural unit that matches the input with a sufficient approximation (as in Growing When Required [2]). The evolving tree [6] is a hierarchical growing neural network with a tree structure and the neural units on each node of the tree. This neural network is attractive because it builds a gross clustering of the input patterns and then go on using a splitting procedure during the training phase. A neuron of the network is split if it is the best matching unit too often, meaning that its cluster is too populated. The network growing is not limited during learning.

This spitting and learning procedure builds a natural hierarchy of clusters that can be easily exploited. The training procedure of this network requires more time as the number of neurons increases, a drawback that can be serious with complex datasets (high dimensional patterns or huge amount of data).

In this paper we present an application of this network to the image segmentation using a faster learning procedure that can overcame the slow training process. Image segmentation is a natural testbed for the speed-up due to the large set of data and it highlights also the performances of the network in clustering. The images are segmented by splitting an initial gross segmentation in a "natural" region splitting procedure.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 356–364, 2007.

The paper is organized as follows: the next subsection explains the evolving tree algorithm, then the algorithm simplification is explained following with some results using some artificial two dimension input distributions, then the image segmentation results are shown and commented.

2 The Evolving Tree Algorithm

The neural units of the Evolving Tree network have two parameters: the weights vector (considered as one parameter) and a counter b_i that counts the times the unit is the winner unit. The learning algorithm is based on a top-down hierarchical process where the input pattern is passed from the root node of the tree to the leaf nodes. When an input vector is presented to an unit it first checks if it is a leaf node or not. If the unit is not a leaf then it calculates the distance of the input pattern from the weight vectors of its children, finds the winner unit, and passes the input to it, then the former steps are repeated. If the node is a leaf of the tree its weights are modified using the formula:

$$w_i(t+1) = w_i(t) + \alpha(t) h_{ci}(t) \left[x(t) - w_i(t) \right] \tag{1}$$

and the winner unit counter b_i is increased.

When the b_i parameter reaches a threshold θ the node is split into fixed number of children units. After the splitting the split neural unit will not learn anymore, because only the leaf nodes of the tree are corrected, this is a behavior that prevents a unit to learn too much [5] and resembles the "Conscience" mechanism [7]. This also means that a tree node is frozen after it was θ times the winning unit.

In the original algorithm the weight correction is propagated to the unit neighborhoods on the tree according to the winner take most principle. The neighborhood function is the usual gaussian neighborhood:

$$h_{ci}(t) = \exp\left(-\frac{\|r_c - r_i\|^2}{2 * \sigma^2} \right) \tag{2}$$

and the distance between unit c and unit i $\|r_c - r_i\|$ is calculated "on the tree" counting the number of "hops" from the winner leaf unit c to the other leaf units of the tree, as described in the original papers [3], [6], [5].

Each network unit is trained for θ steps at most, and considering the usual time decay of $\alpha(t)$ learning parameter reported in eq.3, units that are deep in the hierarchy receive a minor correction:

$$\alpha(t) = \alpha_{max} \left(\frac{\alpha_{min}}{\alpha_{max}} \right)^{\frac{t}{t_{max}}} \tag{3}$$

.

In eq.3 t_{max} is the total number of learning steps.

3 Considerations About the E-Tree Algorithm

There are two considerations about the evolving tree that are worthwhile. The first one is related to the role of the neighborhood function, the second one is related to the learning parameter α and its time dependency.

As in all self–organizing networks the neighborhood function in eq. 2 is used to train the neural units that are near the winner unit. Units that are near each other on the tree can be children of different nodes, for example, according to the original paper, if B_1 is the winner unit then also A_1 and A_2 are updated using the neighborhood function. This means that A_1 and A_2 are "moved" toward B_1 in the input space (see fig 3), and this movement can move A_1 and A_2 outside the Voronoi region V_A of unit A, and toward the Voronoi region V_B of B. Due to this mechanism the winner unit obtained using the top-down search on the tree structure can bring to a unit that is not the global best matching unit as defined in:

$$bmu = \arg\left\{\min_{i \in N}\{\|x - w_i\|\}\right\} \tag{4}$$

This problem was highlighted in [5] where the comparison between global best matching unit calculated with the formula above and the winner unit obtained using the top-down search on the Evolving Tree are compared.

In self–organizing networks the neighborhood function has many purposes, the main two of them are: allowing the spread of the lattice in fixed topology networks, as in the SOM network [1], and ease the unit distribution in a network with a fixed number of neural units, as in Neural Gas [9]. Neighborhood function allows to have more neural units where the input patterns are more frequent.

In Evolving Tree network there is not a structure to organize and, from this point of view, it resembles the Neural Gas.

Moreover the learning algorithm of the Evolving Tree is organized in a *"learn–then–split(if necessary)"* sequence so that all the units are trained before split-

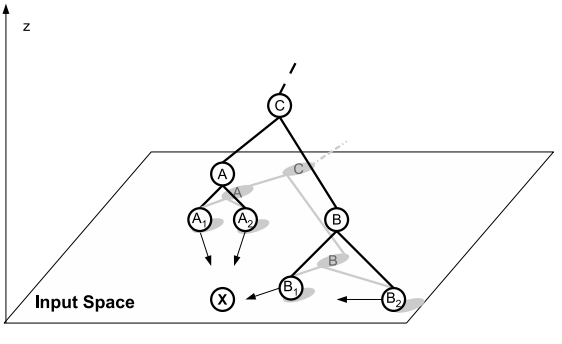


Fig. 1. Correction of the neighborhood units in Evolving Tree network: x is the input pattern. Weight vectors of the neural units are vectors in the input space but, in order to highlight the tree structure another dimension z, orthogonal to the input space, was added.

ting. The splitting procedure generates new units where input patterns are dense. For this reason there are not "dead units" and correction propagation has a little influence on the efficiency of the network. According to this observation, neighborhood function can be neglected reducing the algorithm complexity.

The learning parameter α in eq. 1 decays during the training session as indicated in eq. 3. Due to the nature of the algorithm that creates new units where there is an high input density, we thought that there is not need to have a learning parameter variable with time. Adding more units in areas where there are many input patterns the unit correction will automatically decay because there will be more neurons in the same input space area.

4 Evolving Tree Algorithm Modifications

Due to the considerations of the preceding section, the neighborhood function $h_{ci}(t)$ seems not to be necessary, so that it can be reduced to the following:

$$h_{ci}(t) = \begin{cases} 1 \text{ if } i = c \text{ (i.e. unit i is the winning)} \\ 0 \text{ otherwise} \end{cases} \tag{5}$$

and the algorithm became similar to the hard competitive learning; moreover even the learning parameter α can be a fixed value, α_{const}.

The proposed algorithm modifications, training with constant learning parameter α and /or without neighborhood function $h_{ci}(t)$, are tested using the four non uniform distributions in fig. 2, and the four combinations are summarized in table 1.

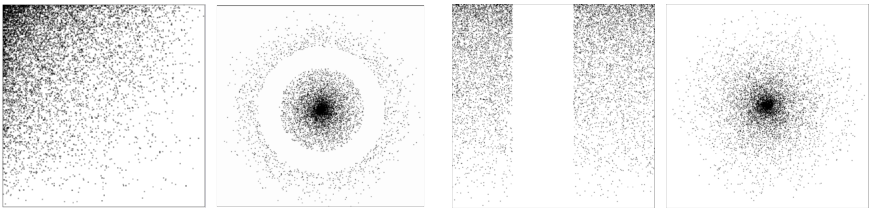
We implemented the learning algorithm in [6], not the version in [5]; for all the experiments the topology is a binary tree and the splitting threshold is $\theta = 400$. If α is variable the values are $\alpha_{max} = 0.5$, $\alpha_{min} = 0.05$, where α is constant is $\alpha_{const} = 0.3$.

For each distribution 10 training session are performed: the first training session is made of $t_{max} = 50.000$ learning steps, the second is made of $t_{max} = 100.000$ training steps and so on to $t_{max} = 500.000$. It should be noticed that due to the growing nature of the algorithm the number of neurons is slightly different from a training session to another but the differences can be neglected.

Fig. 3 shows that without the neighborhood function the network implementation is faster than the original one and the processing time is linear with the number of units.

Table 1. Algorithm modifications

Condition label	α	$h_{ci}(t)$
original algorithm	eq. 3	eq. 2
hard competitive learning	eq. 3	eq. 5
soft competitive learning, constant learning rate	α_{const}	eq. 2
hard competitive learning, constant learning rate	α_{const}	eq. 5



(a) (b) (c) (d)

Fig. 2. Input distributions used for the algorithm evaluation

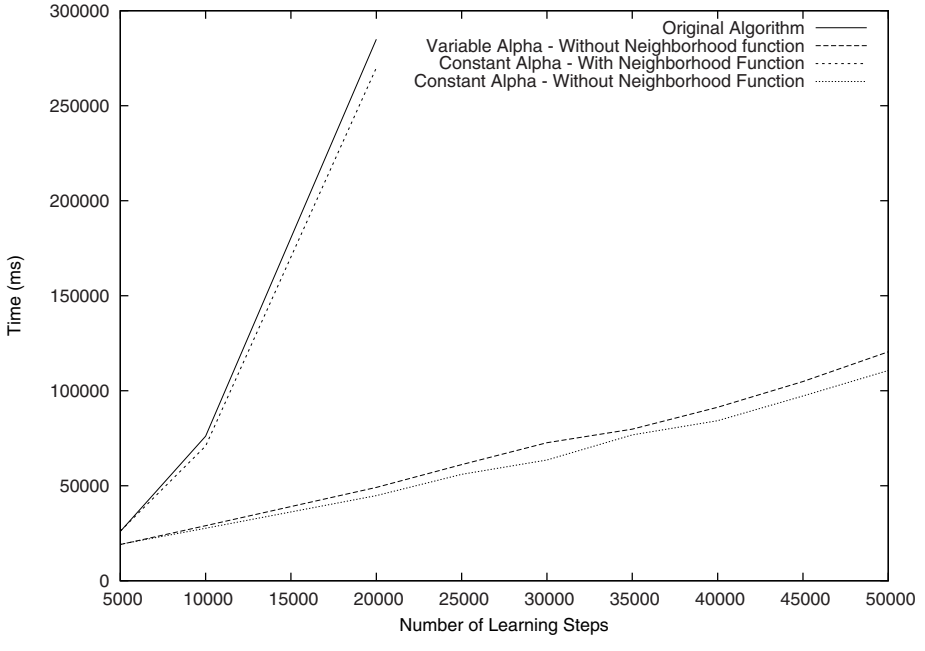


Fig. 3. Mean results for the distributions in fig. 2: the graphs show the processing time values obtained

In order to compare the results obtained from the algorithms two parameters were used: the entropy and the winning unit calculation.

If each unit is considered as a code vector of a codebook, the entropy maximization ensures that the quantization intervals are used with the same frequency during the quantification of the input signal. If the Evolving Tree network has N neural leaf units the input manifold is divided into V_i $i = 1, 2, \ldots N$ regions. After the learning phase, the probability that the input pattern v falls in the interval V_i should be $p(V_i) = \frac{1}{N}$, the entropy is calculated as

$$H = -\sum_{i=1}^{N} p\left(V_i\right) * \log\left[p\left(V_i\right)\right] \tag{6}$$

and the maximum theoretical value will be:

$$H_{max} = \log(N). \tag{7}$$

So using the entropy value, calculated at the end of the learning phase, it is possible to evaluate the distribution of the leaf node of the Evolving Tree in the input space.

Using the different learning algorithm reported in table 1 gives different number of neurons N and different values of theoretical entropy (e.g. eq. 7). The difference on number of neural units is below 4% and leads to a difference in entropy values of 0.8%. So that we consider the theoretical entropy value of the original algorithm as a reference for all the learning algorithm variations in table 1.

The comparison between the winning unit and the $b.m.u$ was used in [5] because the winning unit calculation using the top-down search on the Evolving Tree may not indicate the true best match unit obtained with the global search in the set of neural units. This method was already used in the same paper to evaluate the impact of some perturbations on the learning algorithm.

We report the number of times the global $b.m.u.$ obtained using the eq. 4 is coincident with the winning unit, compared to the total number of learning steps:

$$\frac{b.m.u. \equiv winning\ unit}{t_{max}} \tag{8}$$

the theoretical value should be 1 in all conditions.

The results obtained for different algorithms are compared in fig. 5 and in fig.4. Fig. 5 shows how many times the winning units is the global best matching unit. Fig. 4 shows the entropy values obtained for the original algorithm and the

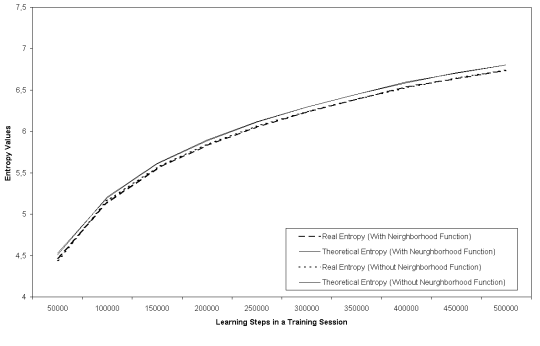


Fig. 4. Mean results for the distributions in fig. 2: the graphs show the comparison of the entropy values obtained

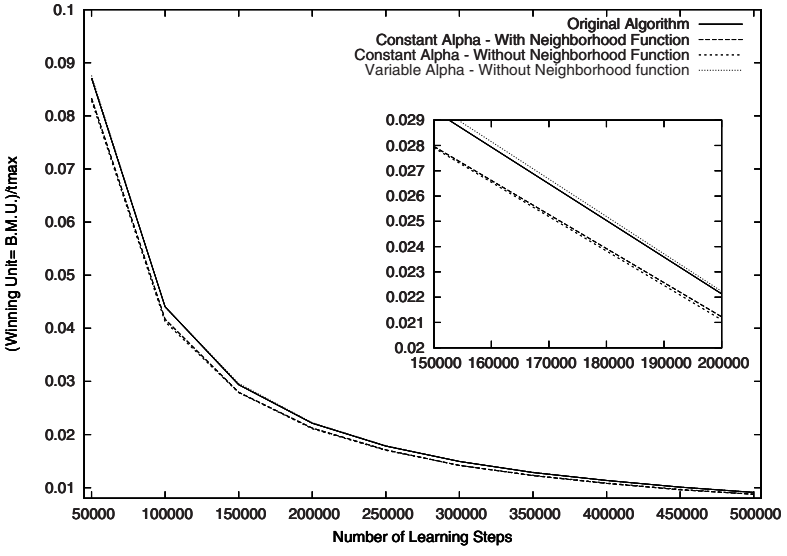


Fig. 5. Mean results for the distributions in fig. 2: the graphs show the number of times the winner unit obtained using the top-down research in Evolving Tree algorithm is the same of the global best matching unit

simplified algorithms. Differences due to the different neurons number can be neglected as said above.

Figures show that with a constant alpha the performances of the network are slightly worse.

5 Image Segmentation Application

In order to obtain the image segmentation the network is trained using a set of pattern X where each pattern $x \in X$is one for each pixel of the image. The input pattern x is obtained using the pixel position (i, j) and the color parameters H and S as shown below

$$x = [c_1 * i, c_1 * j,\ c_2 * H,\ c_2 * S]^T \tag{9}$$

where $c_1 = 0.1$ and $c_2 = 0.8$ are two constants used to weight the pixel position and the color information.

After the learning on an image for segmentation the Evolving Tree builds a hierarchy as in fig. 6. The root of the network receives the image as input, the unit is split in n children ($n = 5$ in fig. 6) and each children unit clusters the image pixels as shown in level 1 image. This is repeated for each unit at all the levels. In figs **??** and **??** on the left is shown the original image and on the right the segmentation results. The four segmented images are obtained on different levels of the network, as fig. 6 shows. The segmentation results in fig. 6 were

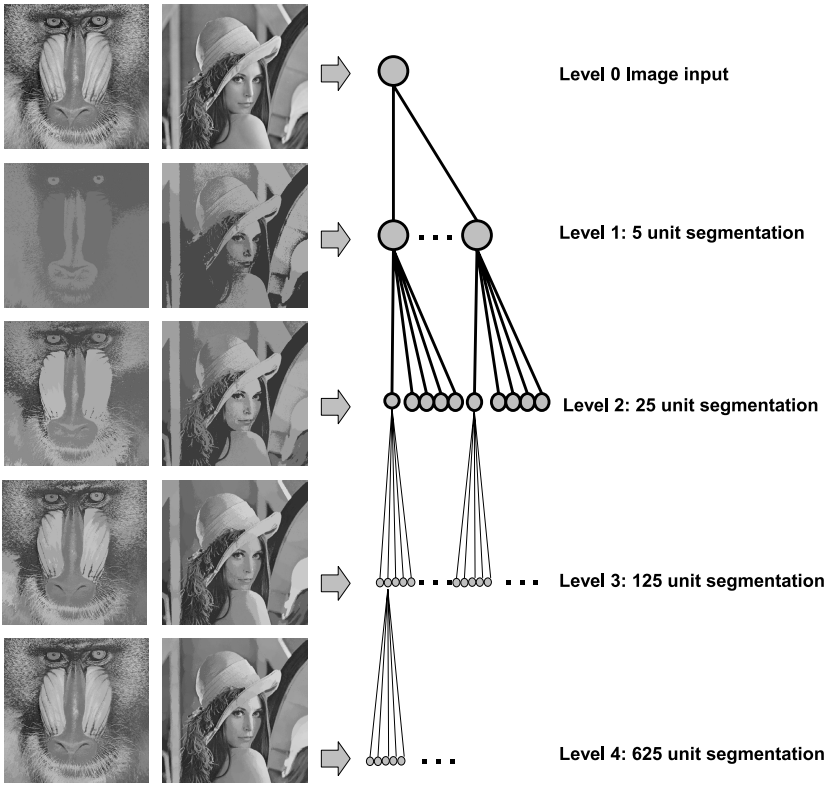


Fig. 6. The segmentation procedure

obtained using the following parameters values : $\theta = 200$, $\alpha = 0.3$, 5 children for each node when splitting, and a learning time $t_{max} = 30000$. For the image Lena in fig. 6 where used the same parameters. The results of the image segmentation are on the left side of fig. 6.

6 Conclusions

The evolving Tree algorithm is an interesting and flexible growing algorithm inspired by the self–organizing map. The results presented demonstrates that it is possible to modify the algorithm with a small or null loss of performances. So the Evolving Tree can be adapted to the specific problem: for example neighborhood function can be neglected in order obtain a speed-up and the learning parameter can be a fixed value in order to use the network with non-stationary input distributions. The faster algorithm is useful when learning with complex data as images or videos. The image segmentation is a natural region splitting and hierarchy structure of the results obtained needs further investigations.

References

1. Kohonen, T.: Self Organizing Maps. Springer, Heidelberg (1997)
2. Marsland, S., Shapiro, J., Nehmzow, U.: A Self-Organizing Network that Grows When Required. Neural Networks 15, 1041–1058 (2002)
3. Pakkanen, J.: The Evolving Tree, a new kind of self-organizing neural network. In: Proceedings of the Workshop on Self-Organizing Maps 2003, pp. 311–316 (2003)
4. Pakkanen, J., Iivarinen, J.: A Novel Self-Organizing Neural Network for Defect Image Classification. In: Proceedings of IJCNN, pp. 2553–2556 (2004)
5. Pakkanen, J., Iivarinen, J., Oja, E.: The Evolving Tree, a Hierarchical Tool for Unsupervised Data Analysis. In: Proceedings of IJCNN, pp. 1395–1399 (2005)
6. Pakkanen, J., Iivarinen, J., Oja, E.: The Evolving Tree — A Novel Self-Organizing Network for Data Analysis. Neural Processing Letters 20, 199–211 (2004)
7. DeSieno, D.: Adding a conscience to competitive learning. In: ICNN 1988. Proc. International Conference on Neural Networks, pp. 117–124. IEEE Service Center, Piscataway, N.J. (1988)
8. Fritzke, B.: A growing neural gas network learns topologies. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) Advances in Neural Information Processing Systems, pp. 625–632. MIT Press, Cambridge (1995)
9. Martinetz, T.M., Berkovich, S.G., Schulten, K.J.: Neural Gas Network for Vector Quantization and its Application to Time-Series Prediction. IEEE Trans. on Neural Networks 4(4), 558–569 (1993)

A Linear Learning Method for Multilayer Perceptrons Using Least-Squares

Bertha Guijarro-Berdiñas, Oscar Fontenla-Romero,
Beatriz Pérez-Sánchez, and Paula Fraguela

Department of Computer Science
Facultad de Informática, Universidad de A Coruña,
Campus de Elviña s/n, 15071 A Coruña, Spain

Abstract. Training multilayer neural networks is typically carried out using gradient descent techniques. Ever since the brilliant *backpropagation* (BP), the first gradient-based algorithm proposed by Rumelhart et al., novel training algorithms have appeared to become better several facets of the learning process for feed-forward neural networks. *Learning speed* is one of these. In this paper, a learning algorithm that applies linear-least-squares is presented. We offer the theoretical basis for the method and its performance is illustrated by its application to several examples in which it is compared with other learning algorithms and well known data sets. Results show that the new algorithm upgrades the learning speed of several backpropagation algorithms, while preserving good optimization accuracy. Due to its performance and low computational cost it is an interesting alternative, even for second order methods, particularly when dealing large networks and training sets.

1 Motivation

Among the many variants of neural network architectures that exist, feed-forward neural networks (and specially, those based on the MultiLayer Perceptron, MLP), are one of the most popular models with successful applications in many fields. The power of these networks comes from having several layers of adaptive weights and nonlinear activation functions (e.g. the sigmoid or hyperbolic tangent). Generally, the sum-of-squares error function is employed for estimating the performance of the network, that compares the desired signal with the network's output. There is not a closed-form solution to find the weight values that minimizes the sum-of-squares error function [9]. Hence the common approach is to use the derivatives of the error function with respect to the weight parameters in gradient-based optimization algorithms for finding the minimum of the error function.

Ever since the first gradient-based algorithm, the brilliant backpropagation (BP) proposed by Rumelhart et al. [1], researchers have focused their efforts on improving the convergence properties of BP, the main concern being the slow convergence speed due to its gradient-descent nature. Some of the newly proposed algorithms that try to improve this aspect are modifications of the

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 365–374, 2007.

original BP such as adding a momentum term [1,2], an adaptive step size [3] or using stochastic learning [4].

Others are second order methods that use the second derivatives of the error function. Some of the most relevant examples of these types of methods are the quasi-Newton approaches including the Levenberg-Marquardt [5] and the conjugate gradient algorithms [6]. Second order methods are among the fastest learning algorithms, however due to their computational cost they are not feasible for large neural networks trained in batch mode.

At last, it is also possible to find methods based on linear least-squares [7,8,9,10,11]. These methods are mostly based on measuring the error of an output neuron *before* the nonlinear transfer function instead of *after* it, as is the usual approach. Usually, they are recommended as training methods for one-layer networks or as initialization methods for multilayer networks.

Specifically, in [10] a method is described to solve a one-layer non-linear neural network using linear least squares. Also using this solution in [12] an algorithm is proposed that linearly calculates the weights of every layer for a multilayer perceptron. However, in this algorithm layers are solved independently and therefore it can only be used as an initialization method. In this paper, we modified this approach by solving each layer in order, from the output layer to the input layer, in such way that the weights in each layer are solved by taking into account the new weights calculated for the succeeding layers. As a consequence, we developed a learning algorithm that improves the learning speed of the basic backpropagation in several orders of magnitude, while maintaining its optimization accuracy.

The organization of this paper is as follows. First, in section 2, we present the method for learning the weights of the network. In Section 3 we investigate the performance of the proposed method on benchmark classification problems and it is compared with several other well-known training methods. In Section 4 these results are discussed. Finally, section 5 bids some suggestions and conclusions.

2 The Proposed Algorithm

In this research we will consider, without loss of generality, a two-layer MLP like the one shown in Fig. 1. The variable names are described below.

Constants I, K, J and S symbolize respectively, the number of inputs, hidden units, outputs and training samples. Each layer of the network consists of a linear matrix $\mathbf{W}^{(n)}$ of weights $w_{ji}^{(n)}$ connecting neuron j in layer n with neuron i in layer $n-1$, thus the superscript $n = 1, \ldots, N$ is used to refer to each layer. These weight matrices are followed by nonlinear mappings $f_j^{(n)}$, regularly selected to be sigmoid-type functions.

For each layer n, the input vectors of the MLP are represented as $\mathbf{x}^{(n)}$. The bias of each layer has been included into weight matrix by adding constant inputs $x_{0s}^{(n)} = 1, \forall n$.

In addition, for all $j = 1, ..., J; s = 1, ..., S$, we will denote by y_{js} the real output obtained by the network, z_{js} the inputs to the non-linearities of the output layer and by d_{js} the desired response provided in the training set. Finally,

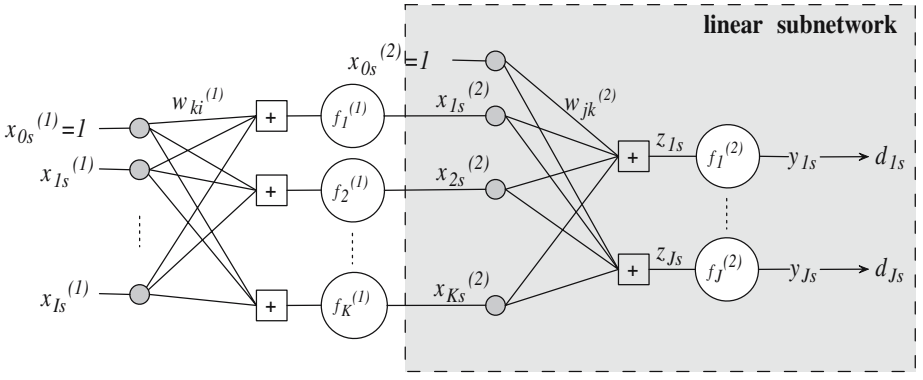


Fig. 1. Architecture of a two-Layer MLP

in the following we will consider as the training optimization criterion, the MSE between the real $\mathbf{y}$ and the desired output $\mathbf{d}$.

2.1 One-Layer Linear Learning: Determining the Weights

Take, for instance, the one-layer neural network corresponding to the shadowy part of Fig. 1. In [10,11,12], the authors considered the approximate least squares optimization of an one-layer nonlinear network assuming the MSE before the nonlinearity as the criterion to be optimized by means of the following theorem (see the proof in [12]),

Theorem 1. *Minimization of the MSE between* $\mathbf{d}$ *and* $\mathbf{y}$ *at the output of the nonlinearity* f *is equivalent (up to first order) to minimizing a MSE between* $\mathbf{z}$ *and* $\bar{\mathbf{d}} = f^{-1}(\mathbf{d})$*, where the inverse function is evaluated at each entry separately. Mathematically, this is given by*

$$\begin{aligned} &\min_{\mathbf{W}} \sum_{s=1}^{S} \sum_{j=1}^{J} \left(f_j \left(\sum_{k=0}^{K} w_{jk} x_{ks} \right) - d_{js} \right)^2 \approx \\ &\min_{\mathbf{W}} \sum_{s=1}^{S} \sum_{j=1}^{J} \left(f_j'(\bar{d}_{js}) \left(\sum_{k=0}^{K} w_{jk} x_{ks} - f_j^{(-1)}(d_{js}) \right) \right)^2 \end{aligned} \tag{1}$$

According to this new error criterion the weights can be optimized by solving a system of $J \times S$ linear equations defined by:

$$\begin{aligned} &\frac{\partial MSE}{\partial w_{jp}} = 2 \sum_{s=1}^{S} \left(f_j'(\bar{d}_{js}) \left(\sum_{k=0}^{K} w_{jk} x_{ks} - f_j^{(-1)}(d_{js}) \right) \right) x_{ps} f_j'(\bar{d}_{js}) = 0; \\ &p = 0, 1, \ldots, K; \quad \forall j \ . \end{aligned} \tag{2}$$

The use of this system presents two main advantages: 1) the global optimum of the training set is obtained, and 2) there is a considerable savings in training time with respect to other gradient-based optimization techniques.

2.2 One-Layer Linear Learning: Determining the Inputs

In [11,12], the authors considered that the weights $\mathbf{W}$ can be fixed, and therefore the input vector $\mathbf{x}$ becomes the free optimization variable. This is a very useful result if $\mathbf{x}$ corresponds to the output (after non linearity) of a hidden layer (see, for example, $\mathbf{x}^{(2)}$ in Fig. 1). Consequently, this result allows the backpropagation of the desired signal for $\mathbf{z}$ to a desired signal for $\mathbf{x}$ through the linear weight layer. The result is summarized in the following theorem, that is proved in [12].

Theorem 2. *Let* $\mathbf{W}$ *be the fixed weight matrix and* $\mathbf{x}$ *the actual input. Then the optimal input* $\mathbf{x}_{opt}$ *that minimizes the MSE between* $\bar{\mathbf{d}} = f^{-1}(\mathbf{d})$ *and* $\mathbf{z}$ *is the input* $\mathbf{x}_{opt}$ *that minimizes a modified MSE criterion between* $\mathbf{x}$ *and the best solution to* $\mathbf{W}\mathbf{x}_{opt} = \bar{\mathbf{d}}$ *in the least squares sense. That is,*

$$\min_{\mathbf{x}} E[(\bar{\mathbf{d}} - \mathbf{z})^T(\bar{\mathbf{d}} - \mathbf{z})] \approx \min_{\mathbf{x}} E[(\mathbf{x}_{opt} - \mathbf{x})^T \mathbf{W}^{\mathbf{T}} \mathbf{W}(\mathbf{x}_{opt} - \mathbf{x})] \tag{3}$$

In this case, the bias is included in the matrix $\mathbf{W}$ *and its corresponding input* (x_0) *is fixed to 1. In this way,* $\mathbf{x}_{opt} = \{x_{1s}, x_{2s}, ..., x_{Ks}\}, \forall s.$

2.3 Combining Theorem 1 and Theorem 2 for Linear Learning of a MLP

Theorem 1 can be used as a basis to provide the desired signal before the non-linearity for every layer of the network and, therefore, linearly find the optimal weights for each layer. Moreover, and in combination with Theorem 2, it is employed to provide a global linear solution for the networks.

The proposed algorithm for linear learning of a multilayer feedforward neural network is as follows:

Step 1: Set the initial weights $\mathbf{W}^{(n)} \forall n$.
Step 2: Using the current weights, propagate the signal forward to calculate the outputs of each layer.
Step 3: Evaluate the value of the MSE between $\mathbf{y}$ and $\mathbf{d}$ and update $\mathbf{W}^{(2)}$ (i.e., the output layer) using the linear system of equations presented in equation 2.
Step 4: Calculate the optimum desired inputs of the output layer (i.e. the desired outputs for the hidden layer) by using the linear system of equations resulting from the right side of equation 3.
Step 5: Update the weights of the hidden layer $\mathbf{W}^{(1)}$ according to the optimal desired outputs calculated in Step 4 and using again the linear system of equations in 2.
Step 6: Check convergence criteria. If they are not reached, continue from Step 2.

The main difference between this algorithm and the proposed one in [12] is the order as the weights of each layer are optimized. In the previous approach first the desired target of all layers is estimated and second the weights are calculated. However, in this case the estimation of the desired output and the calculation of the weights are simultaneously obtained, layer by layer, in a backward way.

3 Experimental Results

In this section the proposed method (linear algorithm) is illustrated by its application to three classification problems of different size, and its performance is compared with four popular learning methods. Two of these methods are of complexity $O(n)$: the gradient descent (GD) and the gradient descent with adaptive momentum and step sizes (GDX), in which the proposed method is based. The other two methods are the scaled conjugated gradient (SCG) (complexity of $O(n^2)$), and the Levenberg-Marquardt (LM) (complexity of $O(n^3)$).

For each experiment all the learning methods shared the following conditions:

- They were carried out in MATLAB® on a 3.20 GHz Pentium 4 processor with 2.5 GB of RAM memory.
- The logistic function was used as the nonlinear functions for neurons.
- The input data set was normalized in the interval [0,1].
- The training process was run for a maximum of 200 epochs.
- For the GDX and the proposed algorithms initial step size was set to 0.005. Moreover the factor used to decrease/increase the learning rate was fixed to 0.01. These values were tuned in order to obtain good results.
- Regarding the topology, a cross-validation method was used to obtain the optimal number of hidden neurons. In this paper only results of the best topology are shown. Specifically for the Breast Cancer and the Wine datasets, 9 and 15 hidden neurons were employed, respectively.
- Each experiment was repeated five times, using a different set of initial weights for each one. This initial set was the same for all the algorithms, and was obtained by the Nguyen-Widrow [13] initialization method.

3.1 Breast Cancer Wisconsin

This two-class problem determines if a patient suffers breast cancer based on several characteristics of the nuclei's cells. The database contains a sample set of 699 instances. The network topology has 9 hidden neurons, 9 inputs and 2 outputs.

Fig. 2 shows the mean MSE training error curves, obtained by each of the tested methods, for the 5 simulations. As can be observed, already in the fourth epoch the proposed method obtains its minimum.

Also, in table 1 some performance measures are shown that allow for the comparison of the algorithms. The first column ($M1$) measures corresponds to the minimum MSE obtained by each method in the training process. The second column ($M2$) is the obtained MSE by each method in the epoch in which the

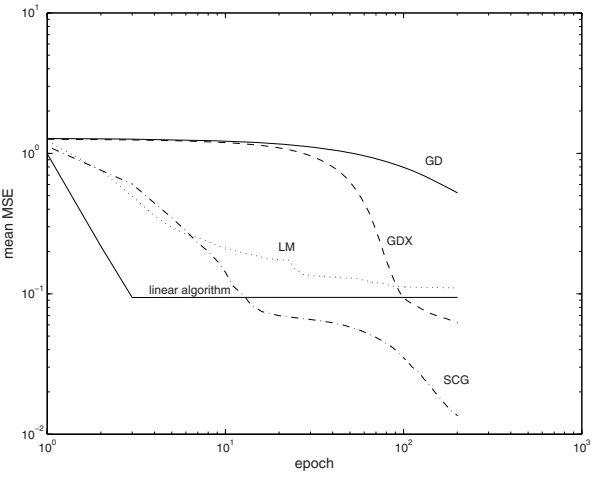


Fig. 2. Mean MSE training error curves for the Breast Cancer Wisconsin dataset

Table 1. Performance measures for the Breast Cancer Wisconsin dataset

Algorithm	M1	M2
GD	$5.22 \times 10^{-1} \pm 2.89 \times 10^{-1}$	$1.26 \pm 3.63 \times 10^{-1}$
GDX	$6.17 \times 10^{-2} \pm 4.76 \times 10^{-3}$	$1.24 \pm 3.98 \times 10^{-1}$
SCG	$1.35 \times 10^{-2} \pm 4.54 \times 10^{-3}$	$4.42 \times 10^{-1} \pm 3.26 \times 10^{-1}$
LM	$1.10 \times 10^{-1} \pm 2.88 \times 10^{-1}$	$3.59 \times 10^{-1} \pm 1.3.99 \times 10^{-1}$
Linear	$9.40 \times 10^{-2} \pm 5.57 \times 10^{-3}$	$9.41 \times 10^{-2} \pm 5.49 \times 10^{-3}$

Table 2. Train and Test Accuracy for the Breast Cancer Wisconsin dataset

Algorithm	Acc. Train (%)	Acc. Test (%)
GD	81.6 ± 2.79	80.3 ± 2.08
GDX	97.7 ± 1.09	$96.7 \pm 2.36 \times 10^{-1}$
SCG	$99.8 \pm 3.99 \times 10^{-2}$	$94.6 \pm 6.90 \times 10^{-1}$
LM	94.5 ± 6.46	87.8 ± 5.21
Linear	$96.0 \pm 3.06 \times 10^{-2}$	$96.0 \pm 2.59 \times 10^{-1}$

proposed method obtained its minimum MSE. As these measures are calculated over the 5 simulations they all are provided in terms of mean and corresponding standard deviation. Moreover, in table 2 train and test classification accuracy, measured at the end of the simulation, are presented.

Finally, table 3 shows both the mean time (in seconds) per epoch $Tepoch_{mean}$ and the mean time of the whole training process $Ttotal_{mean}$ of every algorithm.

Table 3. CPU time comparison (in seconds) for the Breast Cancer Wisconsin dataset

Algorithm	$Tepoch_{mean}$	$Tepoch_{std}$	$Ttotal_{mean}$	$Ttotal_{std}$
GD	1.49×10^{-2}	2.96×10^{-2}	3.081	0.627
GDX	1.40×10^{-2}	2.85×10^{-2}	2.867	0.168
SCG	2.63×10^{-2}	2.99×10^{-2}	5.249	0.628
LM	8.05×10^{-2}	4.52×10^{-2}	15.312	4.988
Linear	4.04×10^{-2}	2.78×10^{-2}	0.349	0.202

3.2 Wine

These data are the results of a chemical analysis of wines growing in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The database includes 178 instances. The network topology has 15 hidden neurons, 13 inputs and 3 outputs.

Fig. 3 shows the mean MSE training error curves obtained by each of the tested methods for the 5 simulations. As can be observed, already in the third epoch the proposed method obtains an error very close to its minimum.

Also, in table 4 again the performance measures $M1$ and $M2$ are shown that allows for the comparison of the algorithms. In this case, 200 iterations are not enough for the other methods to obtain the MSE that the linear algorithm exhibits at the 3rd epoch. Moreover, in table 5 train and test accuracy, measured at the end of the simulation, are also presented.

Finally, table 6 shows the mean time per epoch $Tepoch_{mean}$ and the mean time of the whole training process $Ttotal_{mean}$ of every algorithm.

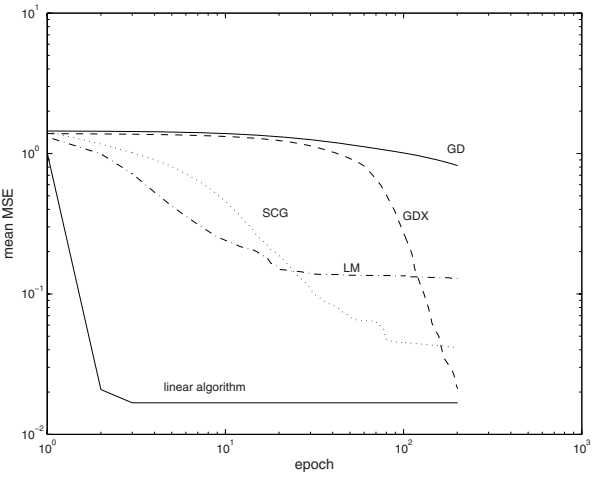


Fig. 3. Mean MSE training error curves for the Wine dataset

Table 4. Performance measures for the Wine dataset

Algorithm	M1	M2
GD	$8.20 \times 10^{-1} \pm 2.76 \times 10^{-1}$	$1.43 \pm 3.33 \times 10^{-1}$
GDX	$2.10 \times 10^{-2} \pm 3.77 \times 10^{-2}$	$1.37 \pm 3.36 \times 10^{-1}$
SCG	$4.16 \times 10^{-2} \pm 1.53 \times 10^{-1}$	$1.01 \pm 2.82 \times 10^{-1}$
LM	$1.29 \times 10^{-1} \pm 2.58 \times 10^{-1}$	$7.18 \times 10^{-1} \pm 4.07 \times 10^{-1}$
Linear	$1.66 \times 10^{-2} \pm 2.18 \times 10^{-3}$	$1.68 \times 10^{-2} \pm 2.23 \times 10^{-3}$

Table 5. Train and Test Accuracy for the Wine dataset

Algorithm	Acc. Train (%)	Acc. Test (%)
GD	49.4 ± 5.32	48.7 ± 9.37
GDX	$99.8 \pm 1.78 \times 10^{-1}$	$97.5 \pm 6.46 \times 10^{-1}$
SCG	96.7 ± 4.71	94.0 ± 3.60
LM	89.4 ± 8.10	86.5 ± 8.63
Linear	$99.9 \pm 3.41 \times 10^{-2}$	$99.0 \pm 2.48 \times 10^{-1}$

Table 6. CPU time comparison (in seconds) for the Wine dataset

Algorithm	$Tepoch_{mean}$	$Tepoch_{std}$	$Ttotal_{mean}$	$Ttotal_{std}$
GD	0.59×10^{-2}	0.77×10^{-2}	1.246	0.145
GDX	0.58×10^{-2}	0.76×10^{-2}	1.182	0.055
SCG	1.25×10^{-2}	0.73×10^{-2}	2.041	0.972
LM	1.48×10^{-1}	5.18×10^{-2}	23.309	12.444
Linear	2.1×10^{-2}	0.82×10^{-2}	0.139	0.143

4 Discussion

From Figs. 2 and 3 and tables 1 and 4 we can see that the proposed method obtains its minimum error at a very early epoch. Regarding its accuracy, it improves the one exhibited by the classical backpropagation and also the Levenberg Marquardt algorithm. Moreover for wine dataset 200 iterations are not enough for the other methods to obtain the minimum MSE that the linear algorithm exhibits.

Also from measure $M2$ in tables 1 and 4 it can be concluded that when our method reaches an error value near its minimum the other algorithms are in a minimum several orders of magnitude higher.

Regarding the accuracy measures shown in tables 2 and 5 it can be deduced that the proposed algorithm again improves the gradient descent and the Levenberg Marquardt algorithms while maintaining an accuracy similar to the GDX and SCG algorithms. It is important to notice that from the tested algorithms it

is the one that shows the more similar accuracy when the results over the train and test sets are compared, thus confirming its generalization ability. Finally, as it can be observed, our method avoids the overfitting since it maintains the same behavior for the training and test sets.

This conclusion, together with the short time needed for training by the proposed method, as shown in tables 3 and 6, definitely makes it a fast and suitable learning algorithm.

5 Conclusions and Future Work

The analyzed results allow us to confirm that the proposed method offers an interesting combination of speed, reliability and simplicity. The method obtains good approximations that even overcome those provided by classic or second order algorithms. These features makes the proposed algorithm suitable for those situations when the speed of the method is important in reaching a good solution, although this could not be always the best one.

Acknowledgements

We would like to acknowledge support for this project from the Xunta de Galicia (project PGIDT05TIC10502PR) and the Ministerio de Educación y Ciencia, Spain (project TIN2006-02402), partially supported by the European Union ERDF.

References

1. Rumelhart, D.E., Hinton, G.E., William, R.J.: Learning representations of back-propagation errors. Nature 323, 533–536 (1986)
2. Vogl, T.P., Mangis, J.K., Rigler, A.K., Zink, W.T., Alkon, D.L.: Accelerating the convergence of back-propagation method. Biological Cybernetics 59, 257–263 (1988)
3. Jacobs, R.A.: Increased rates of convergence through learning rate adaptation. Neural Networks 1(4), 295–308 (1988)
4. LeCun, Y., Bottou, L., Orr, G.B., Müller, K.-R.: Efficient BackProp. In: Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 1524, Springer, Heidelberg (1998)
5. Hagan, M.T., Menhaj, M.: Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 5(6), 989–993 (1994)
6. Beale, E.M.L.: A derivation of conjugate gradients. In: Lootsma, F.A. (ed.) Numerical methods for nonlinear optimization, pp. 39–43. Academic Press, London (1972)
7. Biegler-König, F., Bärmann, F.: A Learning Algorithm for Multilayered Neural Networks Based on Linear Least-Squares Problems. Neural Networks 6, 127–131 (1993)
8. Yam, J.Y.F., Chow, T.W.S, Leung, C.T.: A New method in determining the initial weights of feedforward neural networks. Neurocomputing 16(1), 23–32 (1997)

9. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, New York (1995)
10. Castillo, E., Fontenla-Romero, O., Alonso Betanzos, A., Guijarro-Berdiñas, B.: A global optimum approach for one-layer neural networks. Neural Computation 14(6), 1429–1449 (2002)
11. Fontenla-Romero, O., Erdogmus, D., Principe, J.C., Alonso-Betanzos, A., Castillo, E.: Linear least-squares based methods for neural networks learning. In: Kaynak, O., Alpaydın, E., Oja, E., Xu, L. (eds.) ICANN 2003 and ICONIP 2003. LNCS, vol. 2714, pp. 84–91. Springer, Heidelberg (2003)
12. Erdogmus, D., Fontenla-Romero, O., Principe, J.C., Alonso-Betanzos, A., Castillo, E.: Linear-Least-Squares Initialization of Multilayer Perceptrons Through Backpropagation of the Desired Response. IEEE Transactions on Neural Networks 16(2), 325–337 (2005)
13. Nguyen, D., Widrow, B.: Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In: Proceedings of the International Joint Conference on Neural Networks, vol. 3, pp. 21–26 (1990)

A Discriminative Model Corresponding to Hierarchical HMMs

Takaaki Sugiura, Naoto Goto, and Akira Hayashi

Graduate School of Information Sciences, Hiroshima City University,
3-4-1 Ozuka-higashi, Asaminami-ku, Hiroshima 731-3194, Japan
takaaki@robotics.im.hiroshima-cu.ac.jp

Abstract. Hidden Markov Models (HMMs) are very popular generative models for sequence data. Recent work has, however, shown that on many tasks, Conditional Random Fields (CRFs), a type of discriminative model, perform better than HMMs. We propose Hierarchical Hidden Conditional Random Fields (HHCRFs), a discriminative model corresponding to hierarchical HMMs (HHMMs). HHCRFs model the conditional probability of the states at the upper levels given observations. The states at the lower levels are hidden and marginalized in the model definition. We have developed two algorithms for the model: a parameter learning algorithm that needs only the states at the upper levels in the training data and the marginalized Viterbi algorithm, which computes the most likely state sequences at the upper levels by marginalizing the states at the lower levels. In an experiment that involves segmenting electroencephalographic (EEG) data for a Brain-Computer Interface, HHCRFs outperform HHMMs.

1 Introduction

Hidden Markov Models (HMMs) are very popular generative models for sequence data. Recent work has, however, shown that Conditional Random Fields (CRFs), a type of discriminative model, perform better than HMMs on many tasks [1].

There are several differences between CRFs and HMMs. (1) HMMs are generative models and thus model the joint probability of input (i.e., observations) and output data (i.e., states), whereas CRFs are discriminative models that model the conditional probability of output data given the input data. (2) HMMs make independence assumptions on observations given states, whereas CRFs do not. (3) For model parameter estimation, HMMs do not need the states, whereas CRFs do.

Hierarchical HMMs (HHMMs) are a generalization of HMMs with a hierarchical structure [2]. Murphy [3] has shown that HHMMs are a special kind of Dynamic Bayesian Networks (DBNs), and has derived an efficient inference algorithm [3].

In this paper, we propose the Hierarchical Hidden CRF (HHCRF), a discriminative model that corresponds to the HHMM, a generative model. In hierarchical models, we are mainly interested in the states at the upper levels, and hence,

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 375–384, 2007.

HHCRFs model the conditional probability of the states at the upper levels given observations. The states at the lower levels are hidden, and marginalized in the model definition.

We have developed two algorithms for the model. One is a parameter learning algorithm that needs only the states at the upper levels in the training data. Note that we need all the states to train standard CRF models. The other algorithm is the marginalized Viterbi algorithm, which computes the most likely state sequences at the upper levels by marginalizing the states at the lower levels. Note that a direct extension of the well known Viterbi algorithm computes the most likely joint sequence at the upper and lower levels, which is different from the sequence computed by the marginalized Viterbi algorithm.

In an experiment that involves segmenting electroencephalographic (EEG) data for a Brain-Computer Interface, HHCRFs outperform HHMMs.

2 Related Work

HHMMs were originally defined by Fine *et al.* [2]. Later, Murphy and Paskin [3] devised a DBN representation for HHMMs and a linear time inference algorithm. We have developed a semi-supervised learning algorithm for HHMMs [4]. Applications of HHMMs include handwritten character recognition, information extraction, video structure discovery, and topic transition detection.

CRFs were originally proposed by Lafferty *et al.* [1]. Since then, CRFs have successfully been applied to many problems including parsing, named entity recognition, object recognition, and activity recognition.

Sutton *et al.* [5] proposed Dynamic CRFs (DCRFs), an extension of CRFs, corresponding to factorial HMMs. Liao *et al.* [6] proposed hierarchical CRFs, another extension of CRFs, corresponding to HHMMs. Unfortunately, hierarchical CRFs do not have hidden states, hence all the states must be labeled for model parameter estimation.

Gunawardana *et al.* [7] proposed Hidden CRFs (HCRFs) for phone classification. HCRFs have two advantages over the previous CRF extensions. First, they have output feature functions which can express continuous Gaussian outputs. This is in contrast to binary valued output feature functions for most CRFs. The second advantage is that HCRFs have hidden states. HCRFs have 2 levels. The states at the bottom level are hidden and thus do not need to be labeled for training. The problem with HCRFs is that the states at the top level do not change with time. Therefore, whilst HCRFs can be applied to sequence classification, they cannot be applied to sequence segmentation or sequence labeling.

We were influenced by the above-mentioned approaches, and our HHCRFs share many characteristics with them. In HHCRFs, however, not only are the states at the lower levels hidden, but the states at the upper levels also change with time. This makes it possible to apply HHCRFs to sequence segmentation and sequence labeling without labeling the states at the lower levels for training.

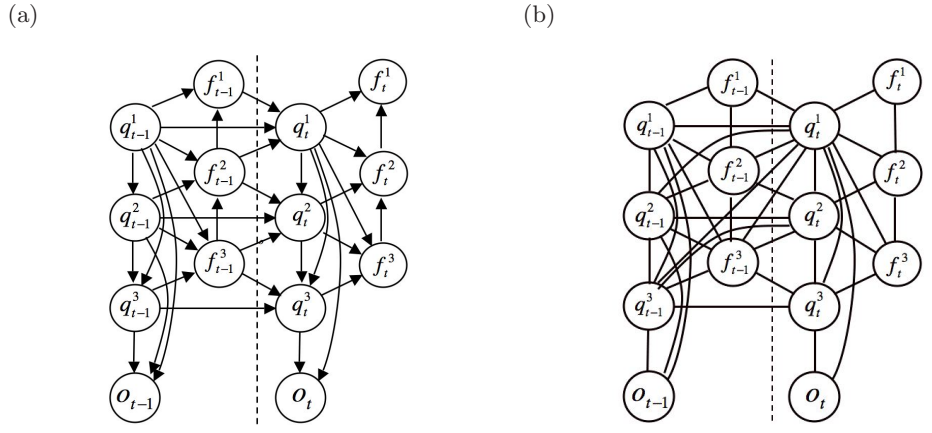


Fig. 1. (a) An HHMM represented as a DBN. (b) An HHCRF represented as an undirected graph. Both (a) and (b) describe only the part of the model from $t-1$ to t.

3 HHMMs

Hierarchical HMMs (HHMMs) are a generalization of HMMs with a hierarchical structure [2]. HHMMs have three kinds of states: internal, production, and end states. They also have three kinds of transitions: vertical, horizontal, and forced transitions. Murphy [3] has shown that an HHMM is a special kind of DBN, and has derived an efficient inference algorithm [3]. In what follows, we show how to represent an HHMM as a DBN.

3.1 Representing an HHMM as a DBN

We can represent an HHMM as a DBN as shown in Fig. 1(a). (We assume for simplicity that all production states are at the bottom of the hierarchy.) A state of the HHMM is denoted by q_t^d ($d \in \{1, \ldots, D\}$), where d is the hierarchy index: the top level has $d = 1$, and the bottom level has $d = D$.

f_t^d is the indicator variable which is equal to 1, if q_t^d has transited to its end state, otherwise it is 0. Note that if $f_t^d = 1$, then $f_t^{d'} = 1$ for all $d' > d$; hence the number of indicator variables that are equal to 0 denotes the level of the hierarchy we are currently on. The indicator variables play an important role in representing the HHMM as a DBN.

Defined below are the transition and output probability distributions. These complete the definition of the model. When q_t^d has transited to its end state, $f_t^d = 1$. This is the signal that the states at the upper levels can be changed. Furthermore, it is a signal that the next value of q_{t+1}^d should be determined by a vertical transition, instead of a horizontal transition. Formally, we denote these as follows:

$$p(q_t^d{=}j'|q_{t-1}^d{=}j, f_{t-1}^{d+1}=b, f_{t-1}^d=f, q_t^{1:d-1}=i) = \begin{cases} \delta(j,j') & \text{if } b=0 \\ A_i^d(j,j') & \text{if} b=1 \text{ and } f=0 \\ \pi_i^d(j') & \text{if } b=1 \text{ and} f=1 \end{cases}$$

$$p(f_t^d=1|q_t^d=j, q_t^{1:d-1}=i, f_t^{d+1}=b) = \begin{cases} 0 & \text{if } b=0 \\ Ae^d(i,j) & \text{if } b=1 \end{cases} \quad (1)$$

$$\mathrm{E}[o_t|q_t^{1:D}=i] = \mu(i)$$
$$\mathrm{Cov}[o_t|q_t^{1:D}=i] = \sigma^2(i)$$

where the state vector $q_t^{1:d} = \{q_t^1, \ldots, q_t^d\}_{d\in\{1,\ldots,D\}}$ is represented by an integer i (i.e. i is the index for "mega state"). In Eq. (1), we assume the dummy state $q_t^0 = 0$ (i.e. the root state) for notational convenience. We also assume dummy indicator variables $f_0^{2:D} = 1$ and $f_t^{D+1} = 1$ for the first slice and for the bottom level, respectively.

$\delta(j,j')$ is Kroneckers delta. $A_i^d(j,j')$ is the horizontal transition probability into the j'th state (except into an end state) from the jth state at level d. $\pi_i^d(j')$ is the vertical transition probability into the j'th state from the ith state at level d. $Ae^d(i,j)$ is the horizontal transition probability into an end state from the jth state at level d.

μ and σ^2 are the mean vector and covariance , respectively, of the Gaussian distribution of the observations emitted at each time. Note that for simplicity, we use scalar observations and a single Gaussian density output. We could of course, use the vector valued observations and a Gaussian mixture density output.

4 HHCRFs

4.1 Model

HHCRFs are undirected graphical models (as shown in Fig. 1(b)) which encode the conditional probability distribution:

$$p(Q^1|O;\Lambda) = \frac{1}{Z(O;\Lambda)} \sum_{Q^{2:D}} \sum_{F^{1:D}} \exp\left(\sum_{k=1}^{K} \lambda_k \Phi_k(Q^{1:D}, F^{1:D}, O)\right) \quad (2)$$

where $Q^1 = \{q_1^1, \ldots, q_T^1\}$ is the state sequence at the top level[1], $O = \{o_1, \ldots, o_T\}$ is the sequence data (observations) and $\Lambda = \{\lambda_1, \ldots, \lambda_K\}$ is the model parameter. We represent the state sequence $Q^{1:D} = \{Q^1, \ldots, Q^D\}$ and the indicator variable sequence $F^{1:D} = \{F^1, \ldots, F^D\}$. $Z(O;\Lambda)$ is the partition function that ensures that $p(Q^1|O;\Lambda)$ is properly normalized.

$$Z(O;\Lambda) = \sum_{Q^1} \sum_{Q^{2:D}} \sum_{F^{1:D}} \exp\left(\sum_{k=1}^{K} \lambda_k \Phi_k(Q^{1:D}, F^{1:D}, O)\right) \quad (3)$$

$\Phi_k(Q^{1:D}, F^{1:D}, O)$ is a feature function that can be arbitrarily selected.

[1] For simplicity, we assume that only the state sequence at the top level is not hidden. We could of course assume that the state sequences at multiple upper levels are not hidden.

In order to compare the performance of HHCRFs with that of HHMMs, which have a Markov structure in the state sequence, we restrict the feature function as $\Phi_k(Q^{1:D}, F^{1:D}, O) = \sum_{t=1}^{T} \phi_k(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t)$ and make the model structure equivalent to that of the HHMMs. Each feature function $\phi_k(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t)$ is as follows.

$$\phi_{j,j',i,d}^{(Hor)}(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t) = \Big(\delta(q_{t-1}^{d} = j) \cdot \delta(q_t^{d} = j') \cdot \delta(q_t^{1:d-1} = i) \cdot \delta(f_{t-1}^{d+1} = 1) \cdot \delta(f_{t-1}^{d} = 0)\Big) \quad \forall_j, \forall_{j'}, \forall_i, \forall_d$$

$$\phi_{i,j',d}^{(Ver)}(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t) = \Big(\delta(q_t^{d-1} = i) \cdot \delta(q_t^{d} = j') \cdot \delta(f_{t-1}^{d+1} = 1) \cdot \delta(f_{t-1}^{d} = 1)\Big) \quad \forall_i, \forall_{j'}, \forall_d$$

$$\phi_{i,j,d}^{(End)}(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t) = \Big(\delta(q_t^{1:d-1} = i) \cdot \delta(q_t^{d} = j) \cdot \delta(f_t^{d+1} = 1) \cdot \delta(f_t^{d} = 1)\Big) \quad \forall_i, \forall_j, \forall_d$$

$$\phi_i^{(Occ)}(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t) = \delta(q_t^{1:D} = i) \quad \forall_i$$

$$\phi_i^{(M1)}(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t) = \delta(q_t^{1:D} = i) \cdot o_t \quad \forall_i$$

$$\phi_i^{(M2)}(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t) = \delta(q_t^{1:D} = i) \cdot o_t^2 \quad \forall_i$$

where $\delta(q = q')$ is equal to 1 when $q = q'$ and 0 otherwise. The first three feature functions are transition features. $\phi_{j,j',i,d}^{(Hor)}$ counts the horizontal transition into the j'th state (except into an end state) from the jth state at level d. $\phi_{i,j',d}^{(Ver)}$ counts the vertical transition into the j'th state from the ith state at level d. $\phi_{i,j,d}^{(End)}$ counts the horizontal transition into an end state from the jth state at level d. $\phi_i^{(Occ)}, \phi_i^{(M1)}, \phi_i^{(M2)}$ are output features which are necessary to represent the Gaussian density output [7].

It can be shown that setting the parameter Λ (i.e., the weight of the feature functions) as follows gives the conditional probability distribution induced by HHMMs with the transition probability distributions and the output probability distributions defined in Eq. (1):

$$\begin{aligned}
\lambda_{j,j',i,d}^{(Hor)} &= \log A_i^d(j, j') & \lambda_i^{(Occ)} &= -\frac{1}{2}\left(\log 2\pi\sigma^2(i) + \frac{\mu^2(i)}{\sigma^2(i)}\right) \\
\lambda_{i,j',d}^{(Ver)} &= \log \pi_i^d(j') & \lambda_i^{(M1)} &= \frac{\mu(i)}{\sigma^2(i)} \\
\lambda_{i,j,d}^{(End)} &= \log Ae^d(i, j) & \lambda_i^{(M2)} &= -\frac{1}{2\sigma^2(i)}
\end{aligned} \tag{4}$$

4.2 Parameter Estimation

Just as in HHMMs, parameter estimation for HHCRFs is based on the maximum likelihood principle given a training set $\mathcal{D} = \{O^{(n)}, Q^{1(n)}\}_{n=1}^{N}$. The difference is that we maximize the conditional probability distribution $p(Q^1|O;\Lambda)$ for HHCRFs, whereas we maximize the joint probability distribution $p(Q^1, O;\Lambda_1)$ for HHMMs. Here, Λ_1 is the parameter for HHMMs. The conditional log-likelihood for HHCRFs is as follows.

$$\begin{aligned}\mathcal{L}(\Lambda) &= \sum_{n=1}^{N} \log p(Q^{1(n)}|O^{(n)};\Lambda) \\ &= \sum_{n=1}^{N} \log \left(\sum_{Q^{2:D}} \sum_{F^{1:D}} \exp \left(\sum_{k=1}^{K} \lambda_k \Phi_k(Q^{1(n)}, Q^{2:D}, F^{1:D}, O^{(n)}) \right) \right) \\ &\quad - \sum_{n=1}^{N} \log Z(O^{(n)};\Lambda) \end{aligned} \tag{5}$$

The gradient of Eq. (5), which is needed for estimating the parameter $\hat{\Lambda}$, is given by

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \lambda_k} &= \sum_{n=1}^{N} \sum_{Q^{2:D}} \sum_{F^{1:D}} \Phi_k(Q^{1(n)}, Q^{2:D}, F^{1:D}, O^{(n)}) p(Q^{2:D}, F^{1:D}|Q^{1(n)}, O^{(n)};\Lambda) \\ &\quad - \sum_{n=1}^{N} \sum_{Q^1} \sum_{Q^{2:D}} \sum_{F^{1:D}} \Phi_k(Q^{1:D}, F^{1:D}, O^{(n)}) p(Q^{1:D}, F^{1:D}|O^{(n)};\Lambda) \end{aligned} \tag{6}$$

The right hand side of Eq. (6) is the difference between the expectation of feature values under the probability distribution $p(Q^{2:D}, F^{1:D}|Q^{1(n)}, O^{(n)};\Lambda)$ and that under $p(Q^{1:D}, F^{1:D}|O^{(n)};\Lambda)$. Since $\Phi_k(Q^{1:D}, F^{1:D}, O) = \sum_{t=1}^{T} \phi_k(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}, o_t)$, the sufficient statistics to compute the first expectation are the transition probabilities $\{p(q_{t-1}^{2:D}, q_t^{2:D}, f_{t-1}^{1:D}, f_t^{1:D}|Q^{1(n)}, O^{(n)};\Lambda)|1 \le t \le T\}$ and the occupancy probabilities $\{p(q_t^{2:D}, f_t^{1:D}|Q^{1(n)}, O^{(n)};\Lambda)|1 \le t \le T\}$. Note that the state sequences are partially labeled because Q^1 is given. These probabilities can be computed using the junction tree algorithm [8], or by converting the hierarchical model to a flat model with mega states and applying the backward-forward-backward algorithm [9]. (We use the latter method in our experiment.) Here, the backward-forward-backward algorithm is an extension of the standard forward-backward algorithm to partially labeled state sequences.

The sufficient statistics to compute the second expectation are the transition probabilities $\{p(q_{t-1}^{1:D}, q_t^{1:D}, f_{t-1}^{1:D}, f_t^{1:D}|O^{(n)};\Lambda)|1 \le t \le T\}$ and the occupancy probabilities $\{p(q_t^{1:D}, f_t^{1:D}|O^{(n)};\Lambda)|1 \le t \le T\}$, which can be computed using the junction tree algorithm, or by converting the hierarchical model to a flat

model with mega states and applying the forward-backward algorithm. (Once again, we use the latter method in our experiment.)

4.3 Marginalized Viterbi Algorithm for HHMMs

The well-known Viterbi algorithm can be used to compute the most likely mega state sequence $[\hat{Q}^{1:D}, \hat{F}^{1:D}] = \mathrm{argmax}_{Q^{1:D},F^{1:D}}\, p(Q^{1:D}, F^{1:D}|O; \hat{\Lambda}_1)$. On the other hand, our marginalized Viterbi algorithm computes the most likely upper level state sequence $[\hat{Q}^1, \hat{F}^2]$ by marginalizing the states at the lower levels.

$$[\hat{Q}^1, \hat{F}^2] = \operatorname*{argmax}_{Q^1,F^2} p(Q^1, F^2|O; \hat{\Lambda}_1) = \operatorname*{argmax}_{Q^1,F^2} \sum_{Q^{2:D}} \sum_{F^{3:D}} p(Q^{1:D}, F^{2:D}, O; \hat{\Lambda}_1) \quad (7)$$

We first explain the marginalized Viterbi algorithm for HHMMs and then for HHCRFs. The algorithm uses dynamic programming to obtain the most likely upper level state sequence $[\hat{Q}^1, \hat{F}^2]$.

Initialize: t=1

$$\delta_1(i) = \log p(q_1^1 = i\ ,\ f_1^2 = 1\ ,\ o_1\ ;\ \hat{\Lambda}_1) \qquad \forall_i$$

Iterate: t=2,...,T

$$\delta_t(i) = \max_{i',1\le\tau<t} \left(\delta_\tau(i') + \log A_0^1(i', i) + \alpha_{\tau,t}(i) \right) \qquad \forall_i$$

$$\psi_t(i) = \operatorname*{argmax}_{i',1\le\tau<t} \left(\delta_\tau(i') + \log A_0^1(i', i) + \alpha_{\tau,t}(i) \right) \qquad \forall_i$$

where

$$\alpha_{\tau,t}(i) = \log p(f_{\tau+1:t-1}^2 = 0\ ,\ f_t^2 = 1\ ,\ o_{\tau+1:t} \mid q_{\tau+1}^1 = i\ ,\ f_\tau^2 = 1\ ;\ \hat{\Lambda}_1)$$

Traceback:

$$\hat{q}_T^1 = \operatorname*{argmax}_i \delta_T(i), \quad t = T$$

$$\text{Iterate while } t > 0: \quad 1)\ (\hat{q}_{t'}^1, t') = \psi_t(\hat{q}_t^1), \quad 2)\ t \leftarrow t'$$

$\delta_t(i)$ is the maximum of the log-probability along a single top level path at time t, that generates the first t observations $o_{1:t}$ and ends in the top level state i. $\psi_t(i)$ contains the traceback information of the highest probability path, and records the previous top level state i' and its ending time τ. Thus, τ is the segmentation boundary time between the sub-sequence $o_{1:\tau}$ ending with the top level state i' and the sub-sequence $o_{\tau+1:t}$ generated by the top level state i. $\alpha_{\tau,t}(i)$ is the probability of sub-sequence $o_{\tau+1:t}$ generated by the top level state i.

The time complexity of the Viterbi algorithm is $O(T)$, whereas that of the marginalized Viterbi algorithm is $O(T^2)$. This is the cost for finding the most likely upper level state sequence.

4.4 Marginalized Viterbi Algorithm for HHCRFs

The marginalized Viterbi algorithm for HHCRFs is similar to that for HHMMs, with $\delta_1(i)$ replaced by $\log \sum_{q_1^{2:D}} \sum_{f_1^{3:D}} \exp\Big(\sum_{k=1}^{K} \lambda_k \phi_k(q_1^{1:D}, f_1^{2:D}, o_1) \cdot \delta(q_1^1 = i) \cdot \delta(f_1^2 = 1)\Big)$, the horizontal transition probability $\log A_0^1(i', i)$ replaced by $\lambda_{i',i,0,1}^{(Hor)}$, and $\alpha_{\tau,t}(i)$ replaced by $\log \sum_{q_{\tau+1:t}^{2:D}} \sum_{f_{\tau+1:t}^{3:D}} \exp\Big(\sum_{t'=\tau+1}^{t} \sum_{k=1}^{K} \lambda_k \phi_k(q_{t'-1}^{1:D}, q_{t'}^{1:D}, f_{t'-1}^{2:D}, f_{t'}^{2:D}, o_{t'}) \cdot \delta(f_\tau^2 = 1) \cdot \delta(f_{\tau+1:t-1}^2 = 0) \cdot \delta(f_t^2 = 1) \cdot \delta(q_{\tau+1}^1 = i)\Big) - \lambda_{0,i,1}^{(Ver)}$. Its time complexity is the same as that for HHMMs: $O(T^2)$.

5 Experiment

We compare the performance of HHCRFs with that of HHMMs in segmenting and labeling EEG time series data for a Brain-Computer Interface (BCI). As reported in literature, the HMM has outperformed Fisher's linear discriminant in a *synchronous* BCI experiment where segmented EEG data is classified [10]. However, HMMs do not perform better than static classifiers in *asynchronous* experiments, where non-segmented continuous EEG data is first windowed and then classified [11,12]. It is difficult to identify the beginning and end of each mental task in asynchronous experiments. We get around this problem by using hierarchical dynamical models and by segmenting and labeling the entire EEG data at the same time without windowing.

In our experiment, we use the BCI Competition III Dataset V, which is characterized as a "multi class problem, continuous EEG" [13]. The data set contains data from 3 subjects during 4 sessions, each 4 minutes long [2]. The subjects perform one of three mental tasks for about 15 seconds and then switch randomly to another task at the operator's request.

Both the HHMMs and the HHCRFs have 2 levels: the top level has 3 states, each corresponding to a mental task, while the bottom level has 5 states with a single Gaussian density output. The parameters of the HHCRFs are initialized using the corresponding parameters of the HHMMs. For hidden top level state inference, which is necessary to estimate the mental task at each time, we use three algorithms: Forward-Backward (FB), Joint Viterbi (JV) [3], and Marginalized Viterbi (MV).

Table 1 shows the accuracy rates for labeling the EEG data (i.e., estimating the mental task) at each time. The data is labeled according to the inferred top

[2] The data from the first 3 sessions is used as training data, whilst the data from the last session is used as test data.

[3] the Viterbi algorithm in the joint space of the top and bottom level states.

Table 1. Accuracy rates for labeling the EEG data. Mean (%) with standard deviation (%) in brackets. Average of 10 runs.

subject	HHMM			HHCRF		
	FB	JV	MV	FB	JV	MV
#1	79.05 (0.00)	78.52 (0.88)	79.54 (0.00)	94.58 (4.80)	80.75 (6.36)	92.77 (5.31)
#2	61.58 (0.00)	52.27 (0.37)	52.07 (0.00)	70.17 (0.08)	51.80 (4.00)	47.15 (0.00)
#3	34.40 (0.00)	34.40 (0.00)	34.40 (0.00)	32.11 (0.00)	32.11 (0.00)	32.11 (0.00)
average	58.34	55.06	55.34	65.62	54.89	57.34

Table 2. Segmenting the EEG data for Subject #1. Top level state changes within ± 0.5 sec. of the true time of change are considered as *true-positive's*. Average of 10 runs.

	HHMM			HHCRF		
	FB	JV	MV	FB	JV	MV
Precision (%)	20.00	16.50	18.18	53.39	23.07	57.32
Recall (%)	37.50	25.00	25.00	52.50	25.00	55.00
F measure (%)	26.09	19.85	21.05	52.92	23.81	56.08

level state at each time. The FB algorithm, which computes the most probable state given all observations, has a high accuracy rate for both HHMMs and HHCRFs [4] .

In Table 2, we evaluate the performance in segmenting the EEG data (i.e., detecting the change of mental task) for Subject #1. The data is segmented when the top level state changes. We can see that HHCRFs outperform HHMMs in segmenting the EEG data. See also Fig. 2.

We also evaluate the performance in estimating the *segment* sequence (i.e., the mental task *order* sequence). During 10 runs, the FB and MV algorithms for HHCRFs produced the correct segment sequence [5] 8 and 6 times, respectively, whereas HHMMs failed to produce the correct sequence.

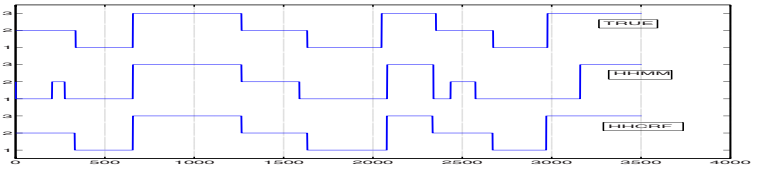


Fig. 2. Top level state sequence for Subject #1 as a function of time. **Top**: true sequence. **Middle**, and **Bottom**: The best HHMM and HHCRF sequences, respectively, in terms of the F measure.

[4] We found the estimation for Subject #3 extremely difficult, as was the case for all the submissions to the competition.

[5] i.e., $2-1-3-2-1-3-2-1-3$.

6 Conclusion

In this paper, we proposed HHCRFs, a discriminative model corresponding to the HHMM. We developed two algorithms for the model: a parameter learning algorithm that needs only the states at the upper levels in the training data, and the marginalized Viterbi algorithm which computes the most likely state sequences at the upper levels by marginalizing the states at the lower levels. In the experiment segmenting EEG data for a Brain-Computer Interface, the HHCRFs outperformed the HHMMs.

References

1. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. 18th Int. Conf. Machine Learning (2001)
2. Fine, S., Singer, Y., Tishby, N.: The hierarchical hidden Markov model: Analysis and applications. Machine Learning 32(1) (1998)
3. Murphy, K., Paskin, M.: Linear time inference in hierarchical HMMs. Advances in Neural Information Processing Systems 14 (2001)
4. Gotou, N., Hayashi, A., Suematsu, N.: Learning with segment boundaries for hierarchical HMMs. In: Proc. 3rd Int. Conf. Advances in Pattern Recognition (2005)
5. Sutton, C., McCallum, A., Rohanimanesh, K.: Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. J. Mach. Learn. Res. 8 (March 2007)
6. Liao, L., Fox, D., Kautz, H.: Hierarchical conditional random fields for GPS-based activity recognition. In: Proc. 12th Int. Symp. of Robotics Research (2005)
7. Gunawardana, A., Mahajan, M., Acero, A., Platt, J.C.: Hidden conditional random fields for phone classification. In: Proc. Int. Conf. Speech Communication and Technology (2005)
8. Huang, C., Darwiche, A.: Inference in belief networks: A procedural guide. Int. J. of Approximate Reasoning 15(3) (1996)
9. Scheffer, T., Decomain, C., Wrobel, S.: Active hidden Markov models for information extraction. In: Hoffmann, F., Adams, N., Fisher, D., Guimarães, G., Hand, D.J. (eds.) IDA 2001. LNCS, vol. 2189, Springer, Heidelberg (2001)
10. Obermaier, B., Guger, C., Neuper, C., Pfurtscheller, G.: Hidden markov models for online classification of single trial eeg data. Pattern Recogn. Lett. 22(12), 1299–1309 (2001)
11. Cincotti, F., et al.: Comparison of different feature classifiers for brain computer interfaces. In: Proc. 1st IEEE EMBS Conference on Neural Engineering, pp. 645–647 (2003)
12. Chiappa, S., Bengio, S.: Hmm and iohmm modeling of eeg rhythms for asynchronous bci systems. In: Proc. European Symposium on Artificial Neural Networks, pp. 199–204 (2004)
13. Blankertz, B., et al.: The bci competition iii: validating alternative approaches to actual bci problems. IEEE Transactions on Neural Systems and Rehabilitation Engineering 14(2), 153–159 (2006)

Finding Unsatisfiable Subformulas with Stochastic Method*

Jianmin Zhang, Shengyu Shen, and Sikun Li

School of Computer Science, National University of Defense Technology
410073 Changsha, China
{jmzhang, syshen, skli}@nudt.edu.cn

Abstract. Explaining the causes of infeasibility of Boolean formulas has many practical applications in various fields. A small unsatisfiable subformula provides a succinct explanation of infeasibility and is valuable for applications. In recent years the problem of finding unsatisfiable subformulas has been addressed frequently by research works, which are mostly based on the SAT solvers with DPLL backtrack-search algorithm. However little attention has been concentrated on extraction of unsatisfiable subformulas using stochastic methods. In this paper, we propose a resolution-based stochastic local search algorithm to derive unsatisfiable subformulas. This approach directly constructs the resolution sequences for proving unsatisfiability with a local search procedure, and then extracts small unsatisfiable subformulas from the refutation traces. We report and analyze the experimental results on benchmarks.

Keywords: Unsatisfiable subformula, Stochastic method, Local search, Resolution sequence, Refutation trace.

1 Introduction

Many real-world problems, arising in artificial intelligence, formal verification and electronic design, can be formulated as constraint satisfaction problems, which are translated into Boolean formulas in conjunctive normal form (CNF). Boolean satisfiability (SAT) solvers are generally able to determine whether a large formula is satisfiable or not. We are usually interested in a small explanation of infeasibility that excludes irrelevant information. Therefore when a formula is unsatisfiable, it is often required to find an unsatisfiable subformula, that is, a small unsatisfiable subset of the original formula. Localizing an unsatisfiable subformula is necessary to determine the underlying reasons for the failure. Explaining the causes of unsatisfiability of Boolean formulas is an essential requirement in many fields. A paradigmatic example is repairing inconsistent knowledge from a knowledge base [1]. Additional examples include SAT-based model checking on predicate abstraction [2], counterexample minimization and explanation [3], and fixing wire routing in FPGAs [4].

* This work is supported by the National Natural Science Foundation of China under grant No. 60603088.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 385–394, 2007.

There have been many different contributions to research on unsatisfiable subformulas extraction in the last few years, owing to the increasing importance in numerous practical applications. Experimental works on computing unsatisfiable subformulas can be grouped into complete search algorithms and incomplete search algorithms. Most of previous works are complete search approaches [5,6,7,8,9,10,11,12,13], essentially on the basis of enhanced versions of the Davis-Putnam-Logemann-Loveland (DPLL) backtrack-search algorithm. In the recent past, a few researches have considered the problem of finding the unsatisfiable subformulas by incomplete methods. In [14], the authors present an algorithm which tracks minimal unsatisfiable subformulas according to the trace of a failed local search run for consistency checking. However, this method is essentially based on a typical local search procedure for giving the formula a satisfiable interpretation. Two distinct algorithms proposed in [15] are the first known works on using stochastic method for proving unsatisfiability of a formula. Whereas up till now, no existing research work has concentrated on the unsatisfiable subformulas extraction from the proof of infeasibility utilizing a stochastic local search procedure.

In this paper, we tackle the problem of extracting unsatisfiable subformulas from refutation traces of Boolean formulas by a stochastic local search algorithm. This approach is the first work we are aware of to adopt resolution-based local search method to find unsatisfiable subformulas. Firstly, a local search procedure is employed to compute the resolution sequences for proving unsatisfiability of a formula. The process of resolving an empty clause is combined with some reasoning heuristics, such as unit clause propagation, binary clause resolution and equality reduction. Then each refutation trace is constructed as a tree, and an effective technique called refutation trace pruning is applied to the tree on-the-fly to reduce the search space. Finally, a recursive function is used to find all of the leaves which correspond to the original clauses, and then a small unsatisfiable subformula is obtained, because the original clauses involved in the derivation of an empty clause are referred to as the unsatisfiable subformula.

The paper is organized as follows. The next section introduces the basic definitions used throughout the paper. Section 3 proposes the local search algorithm for finding small unsatisfiable subformulas. Section 4 presents some heuristics and technique to improve the efficiency of our algorithm. Section 5 shows and analyzes experimental results on well-known pigeon hole problem instances. Finally, Section 6 concludes the paper and outlines future research work.

2 Preliminaries

Resolution is a proof system for CNF formulas with the following inference rule:

$$\frac{(A \vee x)(B \vee \neg x)}{(A \vee B)}\ , \tag{1}$$

where A and B denote the disjunctions of literals. The clauses $(A \vee x)$ and $(B \vee \neg x)$ are the resolving clauses, and $(A \vee B)$ is the resolvent. The resolvent of the

clauses (x) and $(\neg x)$ is the empty clause $(\perp)$. Every application of the inference rule is called a resolution step. A resolution sequence, namely a sequence of resolution steps, is that each one uses the result of the previous step or the clauses of the original formula as the resolving clauses of the current step.

Lemma 1. *A CNF formula φ is unsatisfiable iff there exists a finite sequence of resolution steps ending with the empty clause.*

It is well-known that a Boolean formula in CNF is unsatisfiable if it is possible to generate an empty clause by a resolution sequence from the original clauses. A refutation trace of an unsatisfiable formula is defined as a resolution sequence in which the final resolvent is an empty clause.

Definition 1. ***(Unsatisfiable Subformula).*** *Given a formula φ, ψ is an unsatisfiable subformula for φ iff ψ is an unsatisfiable formula and $\psi \subseteq \varphi$.*

It is obvious that there may exist many different unsatisfiable subformulas with different numbers of clauses for the same problem instance, such that some of these subformulas are the subsets of others.

Lemma 2. *The set of original clauses involved in the derivation of an empty clause is referred to as the unsatisfiable subformula.*

That is to say, the clauses, contained in the intersection of a refutation trace and the original formula, belong to some unsatisfiable subformula. Then we illustrate the process of extracting unsatisfiable subformulas from a Boolean formula according to Lemma 1 and Lemma 2. For example, the CNF formula is

$$\varphi = (x_1) \wedge (\neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3) \ . \tag{2}$$

The formula is refuted by a series of resolution steps ending with an empty clause. There are two resolution sequences to prove the infeasibility of this formula. One of the refutation traces is

$$\frac{(x_1)(\neg x_1 \vee x_2)}{(x_2)} \longrightarrow \frac{(x_2)(\neg x_2)}{(\perp)} \ . \tag{3}$$

From the sequence, the resolvent (x_2) of the first resolution step serves as one of the resolving clauses of the second step, and the result of the second resolution step is an empty clause. According to Lemma 1, this formula is unsatisfiable. Therefore, the original clauses included in the proof of infeasibility belong to an unsatisfiable subformula. More specifically, the unsatisfiable subformula corresponding to the above resolution sequence is

$$\psi_1 = (x_1) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2) \ . \tag{4}$$

Moreover, the other refutation trace is

$$\frac{(x_1)(\neg x_1 \vee x_2)}{(x_2)} \longrightarrow \frac{(x_2)(\neg x_2 \vee x_3)}{(x_3)} \longrightarrow \frac{(x_3)(\neg x_3)}{(\perp)} \ . \tag{5}$$

This finite sequence of resolution steps arrives at an empty clause, and the unsatisfiable subformula consists of the original clauses involved in the process of refutation. Consequently, another unsatisfiable subformula is

$$\psi_2 = (x_1) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3) \ . \tag{6}$$

In conclusion, this simple example demonstrates that our stochastic local search algorithm to find the small unsatisfiable subformulas is essentially based on Lemma 1 and Lemma 2.

3 Local Search for Finding Unsatisfiable Subformulas

In recent years, the complete methods have made great progress in solving many real life problems including constraint satisfaction problem, but they usually cannot scale well owing to the extreme size of the search space. One way to solve the combinatorial explosion problem is to sacrifice completeness, thus some of the best known methods using this incomplete strategy are stochastic local search algorithms. In general, the local search strategy starts from an initial solution, which may be randomly or heuristically generated. Then the search moves to a better neighbor according to the objective function, and terminates if the goal is achieved or no better solution can be found. Stochastic local search methods are underlying some of the best-performing algorithms for certain types of problem instances, both from an empirical as well as from a theoretical point of view. Consequently, this stochastic strategy is adopted to tackle the problem of finding unsatisfiable subformulas. We propose a resolution-based local search algorithm based on Lemma 1 and Lemma 2. The algorithm, detailed in the later, is given as follows:

```
SLSAtoFindUS (formula)
   refuted = false
   iteration = 0
   while ((iteration < MAXITER) && !refuted) do
      if (Unit_Clause_Propagation() return UNSAT) then
         refuted = true
      else if (there exist binary clauses) then
         Binary_Clause_Resolution()
         Non_Tautology()
         Equality_Reduction()
         No_Same_Clause()
      else  Randomly choose two clauses to resolve
      for (each clause c1 added into sequence)
         Trace_Updating(c1)
      if (formula.size > MAXSIZE) then
         Remove a clause c1 at random
         Trace_Pruning(c1)
      iteration++
```

```
if (refuted == true) then
    print unsatisfiable
    SmallUS = Compute_Unsatisfiable_Subformula(sequence)
else  print unresolved
return SmallUS
```

The stochastic local search algorithm begins with an input formula in CNF format. The objective function of this algorithm is to derive an empty clause, and a necessary condition for this to occur is that the formula contains at least some short clauses. We perform resolution of two clauses heuristically or randomly, until the formula is refuted or the upper limit of iterations is reached. In this algorithm, the function called unit clause propagation can determine whether the formula is infeasible, because the formula is refuted if and only if an empty clause can be resolved by two unit clauses. If the current formula contains binary clauses, some reasoning strategies are employed in this algorithm, such as binary clause resolution and equality reduction. The function named Non_Tautology deletes the clauses which contain two opposite literals such as $(x_1 \vee \neg x_1 \vee x_3)$. The function of No_Same_Clause is to remove the duplicate clauses from the formula. If there is no binary clause in the formula, two clauses will be randomly chosen to resolve according to the inference rule shown in Equation 1, and then the resolvent is added into the formula. However, too many resolving clauses increase the overhead of the search process, thus a clause deletion scheme called refutation trace pruning is employed. When the updated formula exceeds the maximum size constant, a clause is removed at random, and some redundant clauses on the source trace of this clause are also deleted.

When the algorithm proceeds, we record the sequences of clauses engaged in the process of resolving an empty clause. Then a tree is constructed with respect to each refutation trace. If the formula is refuted, a recursive function called Compute_Unsatisfiable_Subformula is employed to extract a small unsatisfiable subformula from the formation of a treelike arrangement. According to Lemma 2, we can conclude that all leaf nodes of a tree are actually referred to as the unsatisfiable subformula. Fig. 1 illustrates the process of deriving small unsatisfiable subformulas from the formula denoted by Equation 2. As shown in Fig. 1, there are two trees respectively corresponding to two refutation traces, which are represented as Equation 3 and Equation 5. The original clauses located on the leaves of a tree can be extracted by a recursive algorithm to form the unsatisfiable subformula. For example, in Fig. 1 (a), the root node, namely an empty clause, is resolved by an interim result (x_2) and a leaf node $(\neg x_2)$. If we treat the clause (x_2) as a root node, the branches and leaves with the root also constitute a tree, and then the recursive function can be used on this subtree. The clause (x_2) is resolved by two leaf nodes (x_1) and $(\neg x_1 \vee x_2)$. Consequently, an unsatisfiable subformula is composed of the three leaf clauses belonging to the original formula. Similarly, in Fig. 1 (b), another unsatisfiable subformula consists of the four leaf clauses $(\neg x_3)$, $(\neg x_2 \vee x_3)$, (x_1), and $(\neg x_1 \vee x_2)$.

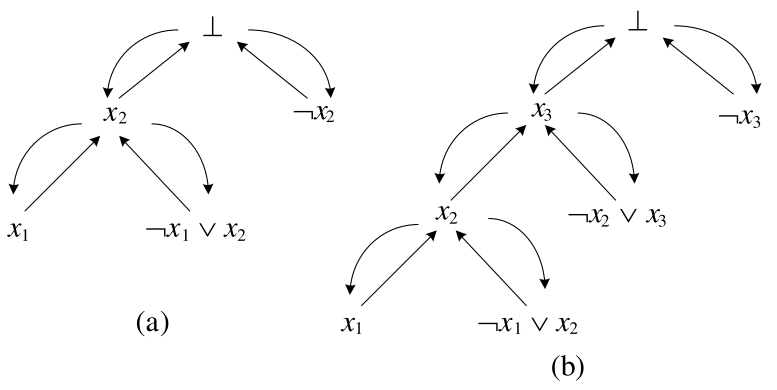


Fig. 1. The Process of Finding Unsatisfiable Subformulas

4 Heuristics and Pruning Technique

To improve the efficiency of the local search algorithm, we implement some reasoning heuristics. One of the heuristics is unit clause propagation. A so-called unit clause is the clause only containing one literal. Unit clause propagation selects a unit clause from the original formula, and then performs the reduction on the formula by this unit clause. We achieve this reduction in two kinds of situation: Firstly, if some clause contains a literal which is negative of the literal in the unit clause, the corresponding literal is deleted from that clause; Secondly, we eliminate the clauses which include the literal of the unit clause. Considered the formula shown in Equation 2, the clause $(\neg x_2)$ is a unit clause, and is propagated to the whole formula. In accordance with the reduction rule of unit clause propagation, the literal (x_2) is removed from the third clause $(\neg x_1 \vee x_2)$, and the fourth clause $(\neg x_2 \vee x_3)$ is deleted. Consequently, the formula is turned into

$$\varphi' = (x_1) \wedge (\neg x_2) \wedge (\neg x_1) \wedge (\neg x_3) \ . \tag{7}$$

After applying the unit clause propagation, one can observe that the formula is strongly simplified and easily refuted. Furthermore, because unit clause propagation might generate new unit clauses, it is an iterative process of executing reductions by unit clauses until either of the following conditions is reached: one is that an empty clause is resolved, or the other is that there are no more unit clauses in the remain formula. The order in which the unit clause reductions occur is not important to the correctness of our local search algorithm.

In general, a Boolean formula might also have many binary clauses, which are defined as the clauses including two literals. Then it is possible to do a lot of reductions on the original formula by reasoning with these binary clauses as well. The resolution of two binary clauses arises if and only if they contain one pair of opposite literals, and abides by the inference rule depicted in Equation 1. For instance, a Boolean formula in CNF is given as follows:

$$\phi_1 = (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_3) \ . \tag{8}$$

There are three binary clauses, which can be resolved by the inference rule, in this formula. Then the process of resolution between the binary clauses is

$$\frac{(\neg x_1 \vee x_2)(\neg x_2 \vee x_3)}{(\neg x_1 \vee x_3)}, \frac{(\neg x_1 \vee x_2)(x_1 \vee x_3)}{(x_2 \vee x_3)}, \frac{(\neg x_1 \vee x_3)(x_1 \vee x_3)}{(x_3)} \ . \tag{9}$$

Resolving these clauses produces two new binary clauses $(\neg x_1 \vee x_3)$, $(x_2 \vee x_3)$ and one new unit clause (x_3). More generally, performing all possible resolutions of pairs of binary clauses may generate new binary clauses or new unit clauses. Therefore, binary clause resolution can be done in conjunction with unit clause propagation in a repeated procedure.

The third heuristic is equality reduction, which is also a type of useful binary clause reasoning mechanism. Equality reduction is essentially based on the following equation:

$$(x \Leftrightarrow y) = (x \Rightarrow y) \wedge (y \Rightarrow x) = (\neg x \vee y) \wedge (\neg y \vee x) \ . \tag{10}$$

If a formula contains two correlated clauses such as $(x \vee \neg y)$ and $(\neg x \vee y)$, we can form an updated formula by equality reduction. Equality reduction is a three-step procedure: Firstly, all instances of y in the formula are replaced by the literal x or vice versa; Secondly, all clauses containing both x and $\neg x$ are deleted; Finally, all duplicate instances of x or $\neg x$ are removed from all of the clauses. For example, a Boolean formula in CNF is

$$\phi_2 = (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee x_3) \ . \tag{11}$$

Obviously, one can conclude that x_1 is equivalent to x_2, because there exist $(x_1 \vee \neg x_2)$ and $(\neg x_1 \vee x_2)$. We substitute x_1 for x_2 throughout the formula, and perform reductions on the new clauses. Then the reduced formula is obtained:

$$\phi_2' = (x_1 \vee x_3) \wedge (\neg x_1 \vee x_3) \ . \tag{12}$$

Similar to binary clause resolution, such clause reasoning approach might yield new binary clauses. Consequently, equality reduction combined with unit clause propagation and binary clause resolution can run iteratively, until an empty clause is resolved or no new clause is added.

During the process of derivation, many redundant clauses bring a degradation of runtime performance and memory consumption. To reduce the search space, we propose a technique called refutation trace pruning, which filters out the clauses not belonging to any refutation proof of the formula. We keep two fields for each interim clause: one is the list of resolving source trace of this clause (*clause.tarce*), the other is a counter that tracks the number of descendants of this clause which still have a chance to involve in the refutation proof (*clause.offspring_count*). This technique contains two functions: one is to establish or update the two fields of trace information when a new clause is added into

the sequence, the other is to remove the clauses which are redundant for proof of unsatisfiability. Firstly, the function called Trace_Updating is introduced:

```
Trace_Updating(c)
    c.trace = resolution_clauses
    c.offspring_count = 0
    for (each clause c1 in c.trace)
        c1.offspring_count++
```

While a clause c is created its *offspring_count* is zero. A newly generated clause can potentially take part in the proof, thus the *offspring_count* of each clause on its resolution trace is incremented. Another function called Trace_Pruning is presented as follows:

```
Trace_Pruning(c)
    if ((c.offspring_count == 0) && (c.trace is not empty))
        for (each clause c1 in c.trace)
            c1.offspring_count--
            Trace_Pruning(c1)
        delete c.trace
```

When a clause c is removed and $c.offpsring\_count \geq 1$, we keep $c.trace$ because we cannot know whether a descendant of c is included in the proof or not. If c has no descendant, the $c.trace$ is deleted and the *offspring_count* for each clause on its resolution source is decremented. These counters may become zero, so a recursive call to the function of Trace_Pruning tries to remove each of the resolution sources.

5 Experimental Results

To experimentally evaluate the effectiveness of our algorithm, we select 9 problem instances from the well-known pigeon hole family, and compare our algorithm with the greedy genetic algorithm [11] on this benchmark. The pigeon hole problem "hole$n$" asks whether it is possible to place $n + 1$ pigeons in n holes without two pigeons being in the same hole. We choose these instances because each of them has only one unsatisfiable subformula. Consequently, the greedy genetic algorithm which derives a minimum unsatisfiable subformula and our algorithm can obtain the same unsatisfiable subformula for one problem instance.

Our algorithm to find small unsatisfiable subformulas is implemented in C++ using STL. The experiments were conducted on a 1.6 GHz Athlon machine having 1 GB memory and running the Linux operating system. The limit time was 3600 seconds. The experimental results are listed in Table 1. Table 1 shows the number of clauses (clas) and the number of variables (vars) for each of the 9 problem instances. Table 1 also gives the number of clauses in the unique

Table 1. Performance Results on Benchmark

Instances	clas	vars	US size	GGA time	SLSA time
hole2	9	6	9	0	0
hole3	22	12	22	0	0
hole4	45	20	45	0	0
hole5	81	30	81	0.02	0
hole6	133	42	133	0.08	0.1
hole7	204	56	204	0.90	0.5
hole8	297	72	297	51.90	22.8
hole9	415	90	415	1304.00	682.6
hole10	561	110	561	time out	1850.0

unsatisfiable subformula (US size) for every instance. Furthermore, Table 1 provides the runtime in seconds of the greedy genetic algorithm (GGA time) to extract the unsatisfiable subformula. The last column presents the mean runtime of ten launches in seconds for the stochastic local search algorithm (SLSA time).

From this table, we may observe the following. The stochastic local search algorithm outperforms the greedy genetic algorithm for most formulas, except for the instance of *hole6*. Our algorithm is able to successfully find the unsatisfiable subformula at each launch. For the instance of *hole10*, the greedy genetic algorithm failed to extract the unsatisfiable subformula within the timeout, but our algorithm succeeded in obtaining it. The resolution-based local search algorithm can efficiently solve these instances, partially because the heuristics brings the great capabilities of reasoning short clauses, such as unit clause propagation, binary clause resolution and equality reduction, and the pigeon hole problem instances contain many binary clauses.

6 Conclusion

Finding the unsatisfiable subformulas of problem instances has practical applications in many fields. In this paper, we present a stochastic local search algorithm to derive small unsatisfiable subformulas from CNF formulas. The algorithm is combined with some reasoning heuristics and pruning technique. The experimental results illustrate that our algorithm outperforms the greedy genetic algorithm. The results also show that this local search algorithm can efficiently tackle the certain type of problem instances with many short clauses, and cannot work very well for the formulas with most long clauses, largely because it makes the decisions on resolution of two long clauses in a stochastic way, and lacks of the effective heuristics for selecting the right clauses. Therefore one of the future works is to add more aggressive methods for resolution of long clauses.

References

1. Mazure, B., Sais, L., Gregoire, E.: Boosting complete techniques thanks to local search methods. Annals of Mathematics and Artificial Intelligence 22(3-4), 319–331 (1998)
2. McMillan, K.L., Amla, N.: Automatic abstraction without counterexamples. In: Garavel, H., Hatcliff, J. (eds.) ETAPS 2003 and TACAS 2003. LNCS, vol. 2619, pp. 2–17. Springer, Heidelberg (2003)
3. Shen, S., Qin, Y., Li, S.: Minimizing counterexample with unit core extraction and incremental SAT. In: Cousot, R. (ed.) VMCAI 2005. LNCS, vol. 3385, pp. 298–312. Springer, Heidelberg (2005)
4. Nam, G-J., Aloul, F., Sakallah, K., Rutenbar, R.: A comparative study of two Boolean formulations of FPGA detailed routing constraints. In: ISPD 2001. Proceedings of the 2001 International Symposium on Physical Design, pp. 222–227 (2001)
5. Zhang, L., Malik, S.: Extracting small unsatisfiable cores from unsatisfiable Boolean formula. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, Springer, Heidelberg (2004)
6. Lynce, I., Marques-Silva, J.P.: On computing minimum unsatisfiable cores. In: Hoos, H.H., Mitchell, D.G. (eds.) SAT 2004. LNCS, vol. 3542, pp. 305–310. Springer, Heidelberg (2005)
7. Oh, Y., Mneimneh, M.N., Andraus, Z.S., Sakallah, K.A., Markov, I.L.: AMUSE: a minimally-unsatisfiable subformula extractor. In: DAC 2004. Proceedings of the 41st Design Automation Conference, pp. 518–523 (2004)
8. Liffiton, M.H., Sakallah, K.A.: On finding all minimally unsatisfiable subformulas. In: Bacchus, F., Walsh, T. (eds.) SAT 2005. LNCS, vol. 3569, pp. 173–186. Springer, Heidelberg (2005)
9. Bailey, J., Stuckey, P.J.: Discovery of minimal unsatisfiable subsets of constraints using hitting set dualization. In: Hermenegildo, M.V., Cabeza, D. (eds.) PADL 2005. LNCS, vol. 3350, pp. 174–186. Springer, Heidelberg (2005)
10. Mneimneh, M.N., Lynce, I., Andraus, Z.S., Marques-Silva, J.P., Sakallah, K.A.: A branch and bound algorithm for extracting smallest minimal unsatisfiable formulas. In: Bacchus, F., Walsh, T. (eds.) SAT 2005. LNCS, vol. 3569, pp. 393–399. Springer, Heidelberg (2005)
11. Zhang, J., Li, S., Shen, S.: Extracting minimum unsatisfiable cores with a greedy genetic algorithm. In: Sattar, A., Kang, B.-H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 847–856. Springer, Heidelberg (2006)
12. Gershman, R., Koifman, M., Strichman, O.: Deriving small unsatisfiable cores with dominator. In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 109–122. Springer, Heidelberg (2006)
13. Dershowitz, N., Hanna, Z., Nadel, A.: A scalable algorithm for minimal unsatisfiable core extraction. In: Biere, A., Gomes, C.P. (eds.) SAT 2006. LNCS, vol. 4121, pp. 36–41. Springer, Heidelberg (2006)
14. Gregoire, E., Mazuer, B., Piette, C.: Tracking MUSes and strict inconsistent covers. In: FMCAD 2006. Proceedings of the the Sixth Conference on Formal Methods in Computer-Aided Design, pp. 39–46 (2006)
15. Prestwich, S., Lynce, I.: Local search for unsatisfiability. In: Biere, A., Gomes, C.P. (eds.) SAT 2006. LNCS, vol. 4121, pp. 283–296. Springer, Heidelberg (2006)

A New Efficient Approach in Clustering Ensembles

Javad Azimi, Monireh Abdoos, and Morteza Analoui

Computer Engineering Department- Iran University of Science and Technology, Tehran, Iran
{Ja_azimi, Abdoos}@comp.iust.ac.ir, Analoui@iust.ac.ir

Abstract. Previous clustering ensemble algorithms usually use a consensus function to obtain a final partition from the outputs of the initial clustering. In this paper, we propose a new clustering ensemble method, which generates a new feature space from initial clustering outputs. Multiple runs of an initial clustering algorithm like k-means generate a new feature space, which is significantly better than pure or normalized feature space. Therefore, running a simple clustering algorithm on generated feature space can obtain the final partition significantly better than pure data. In this method, we use a modification of k-means for initial clustering runs named as "Intelligent k-means", which is especially defined for clustering ensembles. The results of the proposed method are presented using both simple k-means and intelligent k-means. Fast convergence and appropriate behavior are the most interesting points of the proposed method. Experimental results on real data sets show effectiveness of the proposed method.

Keywords: Clustering ensemble, feature space, intelligent k-means, and initial points.

1 Introduction

There is no clustering algorithm performing best for all data sets. Choosing a single clustering algorithm for each data set requires both expertise and insight. Therefore, instead of clustering algorithm, a cluster ensemble can be used [1, 2]. In order to integrate clustering ensembles in a robust and stable manner, one needs a diversity of component partitions for combination that usually obtained from several sources:

1) Using different clustering algorithms to produce partitions for combination [4].
2) Changing initialization or other parameters of a clustering algorithm [3, 5].
3) Using different features via feature extraction for subsequent clustering [1, 6, 7].
4) Partitioning different subsets of the original data [8, 9, 10, 11, 12, 13].

All above introduced mechanisms try to produce more diversity by considering data from different aspects. The major hardship in clustering ensembles is consensus function and partitions combination algorithm to produce final partition, or in the other words finding a consensus partition from the output partitions of various clustering algorithms. The combination of multiple clustering can also be viewed as finding a median partition with respect to the given partitions, which is proven to be NP-complete [14].

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 395–405, 2007.

There are many type of consensus function such as Hypergraph partitioning [1, 6], Voting approach [5, 8, 15], Quadratic Mutual Information Algorithm [16] and Co-association based functions [2, 17, 18]. In this paper, we propose a new consensus function in clustering ensembles, which is named *Labeling Algorithm*. Instead of using previous consensus functions to maintain the results of each k-means and then obtain the final partition using the consensus function, we generate a feature as a result of each k-means run and then run a simple k-means on generated features. In fact, the proposed method can be also classified as a feature extraction method with high precision. Since all of the generated features are the outputs of initial clusterings, each generated feature can classify the samples as accurate as a k-mean algorithm by itself. Therefore, running a clustering algorithm over the generated features can improve the result significantly. Although previous feature extraction methods generate the good features from pure data, each generated feature cannot classify the samples as accurate as simple k-means by itself. Most of the previous studies in clustering ensembles use k-means for initial clustering. It has been reported that the solutions obtained from the k-means are strongly dependent on the initialization of cluster centers [19, 20].

There are many methods which select the initial samples wisely [19, 21, 22] which usually studies the whole feature space to select the initial samples. Since they should study the feature space and select the initial samples using probabilistic method, increasing the time complexity in previous studies is one of the unavoidable problems. In this study, we introduce an intelligent k-means, especially defined for clustering ensembles, which selects the initial samples wisely without any increasing in time complexity. In this paper, the experimental results of the proposed method are presented by both simple k-means and intelligent k-means as initial clustering. The proposed algorithm guarantees that increasing the number of partitions does not decrease the accuracy of clustering ensembles. Sometimes, increasing the number of partitions increases the error rate of the results. Using intelligent k-means instead of simple k-means, increasing the number of partitions usually causes an improvement in results.

The rest of the paper is organized as follows. Section 2 describes the clustering ensembles. The proposed consensus function, named *Labeling Algorithm*, is presented in Section 3. In Section 4, experimental results are presented. We study the complexity of the proposed method beside other methods in Section 5. The paper is concluded in Section 6.

2 Clustering Ensembles

Clustering ensembles usually are two stage algorithms. At the first, they store the results of some independent runs of k-means or other clustering algorithms. Then, they use the specific consensus function to find a final partition from stored results. The major hardship in clustering ensembles is consensus functions and partitions combination algorithm to produce final partition, or in other words finding a consensus partition from the output partitions of various clustering algorithms. There are many types of consensus function such as:

Hypergraph partitioning: The clusters could be represented as hyper edges on a graph whose vertices correspond to the objects to be clustered. The problem of consensus clustering is then reduced to finding the minimum-cut of a hypergraph. The minimum k-cut of this hypergraph into *k* components gives the required consensus partition [1, 6]. Three hypergraph algorithms, CSPA, HGPA, and MCLA, are described in [1] and their corresponding source code are available at http://www.strehl.com.

Voting approach (re-labeling): In the other algorithms, there is no need to explicitly solve the correspondence problem between the labels of known and derived clusters. The voting approach attempts to solve the correspondence problem and then uses a majority vote to determine the final consensus partition [5, 8, 15].

Quadratic Mutual Information (QMI) (also feature-based approach): treats the output of each clustering algorithm as a categorical feature. The collection of L features can be regarded as an ''intermediate feature space'' and another clustering algorithm can be run on it. A mixture model for this case is proposed in [16].

Co-association based functions (also pair wise approach): The consensus function operates on the co-association matrix. Similarity between a pair of objects simply counts the number of clusters shared by these objects in the partitions. Numerous hierarchical agglomerative algorithms (criteria) can be applied to the co-association matrix to obtain the final partition, including Single Link (SL), Average Link (AL), Complete Link (CL) and voting k-means[2,17,18].

3 The Proposed Method

In this paper, we present a new method for clustering ensemble. The proposed method generates a new feature space using the outputs of initial clustering algorithms. The stages of the proposed clustering ensemble are as follows:

1-Using special clustering algorithms to produce initial partitions for combination.
2- Generating new features by *labeling algorithm*.
3- Running a final clustering algorithm on the new generated features.

In the proposed method, both simple k-means and intelligent k-means, introduced in section 3.1, are studied to produce partitions for combination (Stage 1).

Suppose we are given a set of N data points, $X = \{x_1,...,x_N\}$ and a set of H partitions of X return a set of labels for each point $x_i, i = 1,...,N$:

$$x_i \rightarrow \{\pi_1(x_i), \pi_2(x_i)..., \pi_H(x_i)\} \tag{1}$$

Where $\pi_j(x_i)$ denotes a label assigned to x_i by the j-th initial clustering. $\pi_j(x_i)$ is converted into a new value by *labeling algorithm* described in section 3.2.

If y_{ij} denotes to a new label of x_i by j-th initial clustering, we have:

$$y_{ij} = l(\pi_j(x_i)) \ , \ i=1,...,N, \ j=1,...,H \tag{2}$$

Where *l* is the *labeling algorithm* function introduced in Section 3.2. A new feature space is generated by Eq. (2) for each point x_i ,*i=1…,N*. Each clustering algorithm output adds a new dimension in the new feature space. Therefore, each point has *H* components in the new generated feature space, that *H* is the number of initial partitions (Stage 2). A final clustering algorithm is run on the new feature space. We use k-means for both initial and final clustering (Stage 3).

3.1 Intelligent K-Means

It has been reported that the solutions obtaining from k-means are dependent on the initialization of cluster centers [19, 20]. At the first step of the k-means algorithm, we must select *k* initial samples which *k* is the number of clusters. If there are *k* real clusters, then the chance of selecting one sample from each cluster is small. The chance is relatively small when the number of clusters is large. If *k* clusters have equal samples (*n*), then the chance of selection of one sample from each cluster is:

$$p = \frac{\alpha}{\beta} = \frac{k! n^k}{(kn)^k} = \frac{k!}{k^k} \tag{3}$$

Where α , is the number of ways to select one samples from each cluster and β is the number of ways to select *k* samples.

There are many methods which select the initial samples wisely [19, 21, 22]. They studied the whole feature space to select the initial samples and so they increase the complexity.

We propose a new algorithm to refine the initial samples of k-means especially for clustering ensembles without any increasing in complexity. In clustering ensembles, the k-means is run several times. In proposed algorithm, the first execution of k-means uses random initial seed points, but for other executions, we use the previous result of k-means algorithm to select the initial seed points. The initial points for execution *i* are selected from the result of execution *i-1* of k-means algorithm. One sample from each cluster is selected at random as the initial points of next execution of k-means. Therefore, the complexity of the proposed method is $O(1)$.

3.2 Labeling Algorithm

We introduce a new algorithm generating a new feature space based on the k-means outputs with intelligent initial points. Since each generated feature is as accurate as intelligent k-means, the clustering algorithm, which is run over the generated features, is expected to be significantly more accurate than original feature space. In fact, instead of using previous consensus functions to maintain the results of each k-means and then obtain the final partition using the obtained results, we generate a feature as a result of each k-means run and then run a simple k-means on generated features.

The labeling algorithm is run after each execution of k-means algorithm. Graph theory is used in the labeling algorithm. Assume that $G =< V, E >$ denotes a graph obtained by a clustering algorithm output. *G* is a complete weighted graph with *V* vertices and E edges. *V* and *E* represent the cluster centers and edges between every

two cluster centers, respectively. The weight of each edge is the Euclidean distance between two cluster centers.

We consider an approach to generate a spanning tree, $T =< V, E' >$, for a given graph. One of the approaches to generate a spanning tree is to choose a sequence of n-1 edges, in a graph with n nodes. We proposed a greedy algorithm, similar to Prim's algorithm, to generate a spanning tree as follows:

1. Set v_0 as v_i.

2- Select v_j from V which edge_{ij} is smaller than edge_{ik} for each k and set $label(v_j) = label(v_i) + f(w(i, j))$.

3- Remove v_j from V.

4- Set v_j as v_i.

5- Continue from step 2 until all vertices have been selected.

The new generated tree has two nodes of degree one while the degree of the others are two. The obtained spanning tree is used for labeling the samples of each cluster. All samples whose cluster is v_j are labeled with $label(v_j)$.

The proposed labeling method has following characteristics:

1- All samples of each cluster have the same label.
2- The function f is a heuristic function of the edges weight.
3- Different clusters have different labels.
4- The value of α is optional.

For more details on this process, consider an example of a data set $X = \{x_1, ..., x_{20}\}$ with two features assigned to four classes as shown in Fig. 1.a.

The first step in clustering ensembles is generating ensemble members. After each generation the labeling algorithm is run on obtained results.

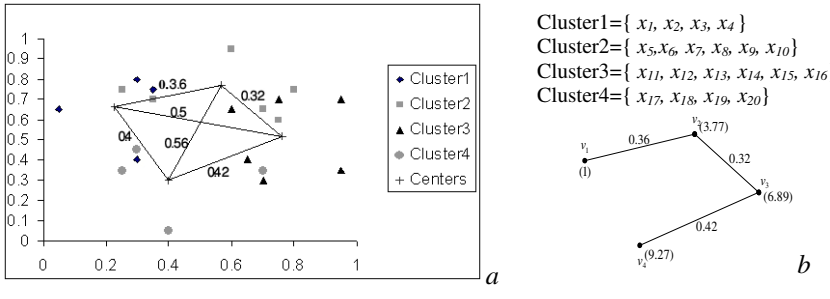


Fig. 1. a) The scatter plot of the samples **b)** Spanning and labeling of the grap

At the first we label the samples in cluster $l(v_1)$ as 1($l(v_1) = 1$). Then the nearest cluster to v_1 is v_2. The label of the samples of cluster v_2 is set by using by

$(l(v_1) + f(w(1,2)))$ that $f(w(i,j)) = 1/w(i,j)$. The nearest cluster to v_2 is v_3 which their samples are labeled as $(l(v_2) + f(w(2,3)))$. At last, the samples of cluster 4 (v_4) are labeled as $(l(v_3) + f(w(3,4)))$ as you can see in Fig.1.b.

The cluster centers are labeled as follows:

$l(v_1) = 1,$ $l(v_2) = l(v_1) + f(w(1,2)) = 1 + 2.77 = 3.77$

$l(v_3) = l(v_2) + f(w(2,3)) = 3.77 + 3.12 = 6.89,$ $l(v_4) = l(v_3) + f(w(3,4)) = 6.89 + 2.38 = 9.27$

The new feature space is generated according to $\pi_1(x_i)$. The output class and the assigned label of each sample are shown in Table 1.

Table 1. The first ensemble member result and assigned label

x_i	Obtained class $\pi_1(x_i)$	Assigned label y_{i1}	x_i	Obtained class $\pi_1(x_i)$	Assigned label y_{i1}
x_1	1	1	x_{11}	2	3.77
x_2	4	9.27	x_{12}	3	6.89
x_3	1	1	x_{13}	3	6.89
x_4	2	3.77	x_{14}	3	6.89
x_5	1	1	x_{15}	3	6.89
x_6	2	3.77	x_{16}	4	9.27
x_7	2	3.77	x_{17}	3	6.89
x_8	2	3.77	x_{18}	4	9.27
x_9	3	6.89	x_{19}	4	9.27
x_{10}	3	6.89	x_{20}	1	1

The proposed mechanism is done after each initial clustering algorithm. Finally, the k-means algorithm is run on the new generated feature space to obtain the final partition.

4 Experiments

The experiments were performed on several data sets, including, four data sets from the UCI repository, “Iris”, “Wine”, “Soybean” and “Thyroid” data sets. A summary of data sets characteristics is shown in Table 2.

Table 2. A summary of data sets characteristics

Name	#of samples	#of features	#of classes	Samples per class
Thyroid	215	5	3	150-35-30
Iris	150	4	3	50-50-50
Wine	178	13	3	59-71-48
Soybean	47	35	4	10-10-10-17

4.1 Heuristic Functions in Labeling

After finding the spanning tree, a heuristic function is used to label the clusters. In experience, we study three different functions for labeling as follows:

$$f_1(w(i,j)) = \frac{1}{w(i,j)} \tag{4}$$

$$f_2(w(i,j)) = w(i,j) \tag{5}$$

$$f_3(w(i,j)) = \frac{w(i,j)}{\arg\min_{i,j} w(i,j)} \tag{6}$$

Tables 4-7 report the mean error rate (%) of clustering combination from 50 independent runs. In tables 4-7, the mean error rate of four different available consensus functions are reported: Co-association function and Average Link (CAL), CSPA and HPGA (which are described in section 2) and the proposed method which is described in this paper with $f_3(w(i,j))$ as a heuristic functions. Parameter H represents the number of partitions and $\alpha = 1$.

In tables 3-6 we can see when the number of partitions is between 10 and 20, we can usually obtain the best results. The error variance value of the proposed method is usually less than the other methods; it means that the proposed method is fast convergence and H-independent compared with the others methods. Another fact is that the proposed method has usually an appropriate accuracy in common data sets.

Table 3. Mean error rate (%) for Iris

H	CAL	CSPA	HGPA	Proposed Method $f = f_3(w(i,j))$
5	12.7	6.38	19.81	8.36
10	9.97	5.23	7.97	4
15	7.73	4.32	5.05	4
20	6.17	4.23	4	4.89
25	5.03	4.3	4	4
30	5.57	4.3	4	4
35	5.07	4.33	4	4
40	5.53	4.27	4	4
45	5.6	4.4	4	4.87
50	5.5	4.5	4	4

Table 4. Mean error rate (%) for Soybean

H	CAL	CSPA	HGPA	Proposed method $f = f_3(w(i,j))$
5	7.02	15.74	18.51	14.04
10	7.01	13.4	15.96	6.38
15	7.52	12.55	14.57	8.09
20	6.55	13.09	14.57	9.08
25	6.88	13.19	15.21	7.94
30	6.21	14.26	15	9.86
35	4.55	14.15	14.47	5.11
40	5.21	13.94	15.11	7.8
45	4.22	13.94	15.85	5.04
50	4.51	13.51	15.85	3.48

Table 5. Mean error rate (%) for Thyroid

H	CAL	CSPA	HGPA	Proposed Method $f = f_3(w(i,j))$
5	24.3	49.35	43.77	12.25
10	20.9	49.26	39.72	12.51
15	9.86	48.6	40.47	13.32
20	7.19	48.47	38.26	13.04
25	16.7	48.88	37.47	12.85
30	15.9	48.6	38.12	13.58
35	16.0	48.84	38.4	13.22
40	16.5	49.09	37.56	13.18
45	16.6	49.14	39.87	13.04
50	15.9	48.65	36.65	13.13

Table 6. Mean error rate (%) for Wine

H	CAL	CSPA	HGPA	Proposed Method $f = f_3(w(i,j))$
5	11.7	10.65	15.98	11.87
10	13.3	9.97	10.06	9.91
15	9.19	10.03	8.57	9.49
20	11.3	10.39	7.42	8.11
25	10.6	10.37	9.04	9.97
30	10.5	10.25	8.15	9.82
35	9.97	10.53	7.39	8.69
40	10.1	10.67	8.09	9.18
45	9.65	10.51	7.84	9.4
50	9.88	10.11	7.53	9.59

The results of simple k-means and intelligent k-means with three different heuristic functions have been shown in Figs 2-5. All of the three functions have approximately the same quality in labeling data. It is so clear that the intelligent k-means has a better

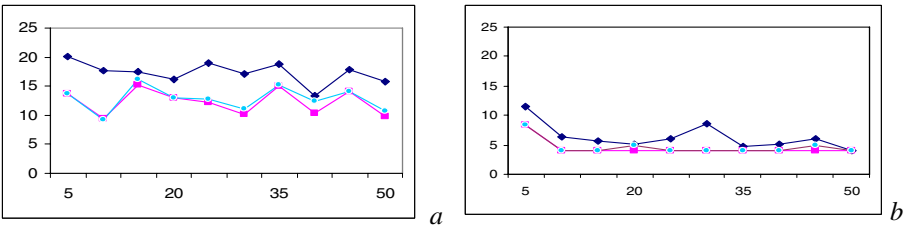


Fig. 2. The results on Iris a)with simple k-means, b) intelligent k-means (α =1)

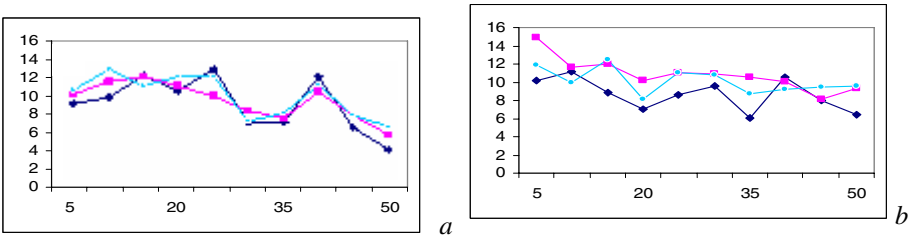


Fig. 3. The results on Wine a)with simple k-means, b) intelligent k-means (α =1)

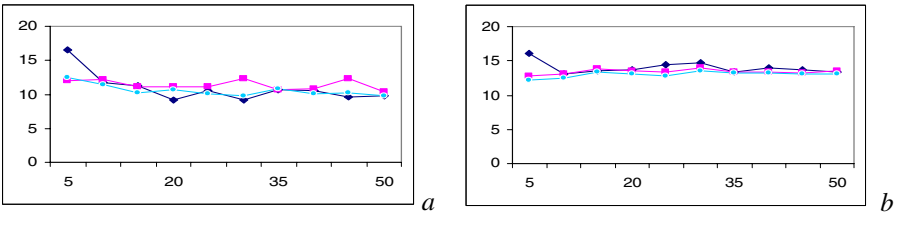


Fig. 4. The results on Thyroid a)with simple k-means, b) intelligent k-means (α =1)

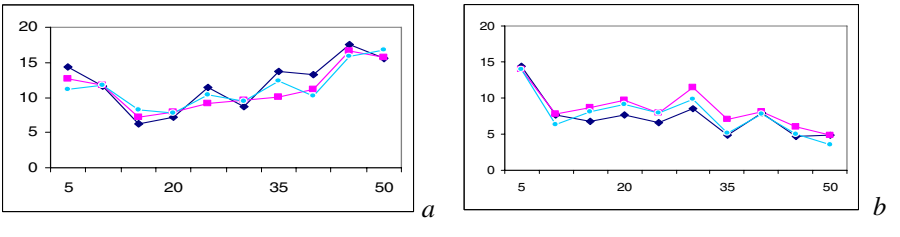


Fig. 5. The results on Soybean a)with simple k-means, b) intelligent k-means (α =1)

behavior than simple k-means in different partition size. In addition, Figure 5.a, on soybean dataset, shows that the results of simple k-means has unexpected variations. But when we select the initial samples intelligently, the unexpected variation can not be seen (Fig. 5.b). An improvement in results is expected by increasing the number of partitions. But left side of Figs 2-5 demonstrates that increasing the number of partitions does not guarantee an improvement in final results. Sometimes, increasing the number of partitions increases the error rate of result (Simple k-means). Right side of Figs 2-5 shows that increasing the number of partitions usually improves results. Although some times increasing the number of partitions does not improve the accuracy of results, it does not decrease the accuracy.

5 Complexities

Time complexity is one of the most important parameters in clustering ensembles algorithms. In this section, we compare the time complexity of different proposed consensus functions.

Hypergraph partitioning: Efficient heuristics to solve the k ways min-cut partitioning problem are known, some with computational complexity on the order of O(|ε|), where ε is the number of hyperedges.

Voting approach: all the partitions from the ensemble must be re-labeled according to a fixed reference partition. The complexity of this process is k!, which can be reduced to O(k^3) if the Hungarian method is employed for the minimal weight bipartite matching problem.

QMI: the complexity of this consensus function is O(kNB), where B is the number of partitions. Though the *QMI* algorithm can be potentially trapped in a local optimum, its relatively low computational complexity allows using multiple restarts in order to choose a quality consensus solution with minimum intra-cluster variance.

Co-association based functions: the computational complexity of co-association based consensus algorithms is very high, O (kN^2d^2).

The proposed method uses k-means for clustering data. The complexity of k-means is $O(kNId)$ where k is the number of clusters and N is the number of samples and I is the number of iterations of k-means to converge in each execution and d is the number of features (dimensions). Therefore, the complexity of the proposed method is $O(k!+kNIdd')$, where, d' is the number of partitions, in the other words, the number of generated features. $k!$ is the complexity time to generate spanning tree. Since k is a small number, $k!$ can be neglected. Therefore, we can see that the complexity of the proposed method is very low.

6 Conclusion

In this paper, we proposed an approach in clustering ensembles. The proposed approach generates a new feature space from the k-means outputs. Each k-means execution generates a new feature. Finally, the k-means algorithm is run on the new generated feature spaces to obtain the final partition.

The complexity of the proposed method is $O(k!+kNIdd')$ where, d' is the number of partitions, in the other words, the number of generated features. An intelligent k-means, which selects the initial samples wisely, has been proposed in this paper. The proposed selecting initial samples algorithm guaranteed that increasing the number of partitions do not decrease the accuracy of clustering ensembles. The complexity of the proposed algorithm for selecting initial points in k-means is $O(1)$. Fast convergence, the novelty and appropriate behaviors are the most interesting points of the proposed method. Using the introduced method, we can make the clustering ensembles as an incremental method, which will be very important in further studies.

References

1. Strehl, A., Ghosh, J.: Cluster ensembles—a knowledge reuse framework for combining partitioning. In: Proc. of 11th National Conf. on Artificial Intelligence, Edmonton, Alberta, Canada, pp. 93–98 (2002)
2. Fred, A.L.N., Jain, A.K.: Data Clustering Using Evidence Accumulation. In: ICPR 2000. Proc. of the 16th Intl. Conf. on Pattern Recognition, Quebec City, pp. 276–280 (2002)
3. Topchy, A., Jain, A.K., Punch, W.: Combining Multiple Weak Clustering. In: Proc. 3d IEEE Intl. Conf. on Data Mining, pp. 331–338 (2003)
4. Hu, X., Yoo, I.: Cluster ensemble and its applications in gene expression analysis. In: Chen, Y.-P.P. (ed.) Proc. 2nd Asia-Pacific Bioinformatics Conference, Dunedin, New Zealand, pp. 297–302 (2004)
5. Fern, X.Z, Brodley, C.E.: Random projection for high dimensional data clustering: a cluster ensemble approach. In: ICML. Proc. 20th International Conference on Machine Learning, Washington, DC, pp. 186–193 (2003)
6. Strehl, A., Ghosh, J.: Cluster ensembles a knowledge reuse framework for combining multiple partitions. Journal on Machine Learning Research, 583–617 (2002)
7. Greene, D., Tsymbal, A., Bolshakova, N., Cunningham, P.: Ensemble clustering in medical diagnostics. In: Long, R., et al. (eds.) Proc. 17th IEEE Symp. on Computer-Based Medical Systems, pp. 576–581 (2004)
8. Dudoit, S., Fridlyand, J.: Bagging to improve the accuracy of a clustering procedure. Bioinformatics 19, 1090–1099 (2003)
9. Fischer, B., Buhmann, J.M.: Bagging for path-based clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1411–1415 (2003)
10. Fred, A.L.N., Jain, A.K.: Robust data clustering. In: CVPR. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, USA, vol. II, pp. 128–136 (2003)
11. Minaei, B., Topchy, A., Punch, W.F.: Ensembles of Partitions via Data Resampling. In: ITCC 2004. Proc. Intl. Conf. on Information Technology, Las Vegas (2004)
12. Monti, S., Tamayo, P., Mesirov, J., Golub, T.: Consensus clustering: a resampling based method for class discovery and visualization of gene expression microarray data. Machine Learning 52, 91–118 (2003)
13. Topchy, A., Minaei-Bidgoli, B., Jain, A.K., Punch, W.: Adaptive Clustering ensembles. In: ICPR 2004. Proc. Intl. Conf on Pattern Recognition, Cambridge, UK, pp. 272–275 (2004)
14. Barthelemy, J.P., Leclerc, B.: The median procedure for partition. In: Partitioning Data Sets. AMS DIMACS Series in Discrete Mathematics, pp. 3–34 (1995)

15. Weingessel, A., Dimitriadou, E., Hornik, K.: An ensemble method for clustering. Working paper (2003), http://www.ci.tuwien.ac.at/Conferences/DSC-2003/
16. Topchy, A., Jain, A.K., Punch, W.: A mixture model for clustering ensembles. In: Proceedings of SIAM Conference on Data Mining, pp. 379–390 (2004)
17. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley & Sons Inc., New York (2001)
18. Aarts, E.H.L., Eiben, A.E., Van Hee, K.M.: A general theory of genetic algorithms. Tech.Rep.89/08, Einndhoven University of Technology (1989)
19. Bradley, P., Fayyad, U.: Refining initial points for k-means clustering. In: Proceedings 15th International Conf., on Machine Learning, San Francisco, CA, pp. 91–99 (1998)
20. Pena, J., Lozano, J., Larranaga, P.: An Empirical comparison of four initialization methods for the k-means algorithm. Pattern Recognition Letters 20, 1027–1040 (1999)
21. Babu, G., Murty, M.: A near optimal initial seed value selection in k-means algorithm using a genetic algorithm. Pattern Recognition Letters 14, 763–769 (1993)
22. Linde, Y., Buzo, A., Gray, R.: An algorithm for vector quantizer design. IEEE trans. Comm. 28, 84–95 (1980)

An Evolutionary Hyperheuristic to Solve Strip-Packing Problems*

Pablo Garrido and María-Cristina Riff

Department of Computer Science, Universidad Técnica Federico Santa María, Valparaíso, Chile
{pgarrido, mcriff}@inf.utfsm.cl

Abstract. In this paper we introduce an evolutionary hyperheuristic approach to solve difficult strip packing problems. We have designed a genetic based hyperheuristic using the most recently proposed low-level heuristics in the literature. Two versions for tuning parameters have also been evaluated. The results obtained are very encouraging showing that our approach outperforms the single heuristics and others well-known techniques.

Keywords: Hyperheuristic, Strip Packing, Evolutionary Algorithms.

1 Introduction

In this paper we focus our attention on methods to solve the two-dimensional strip packing problem, where a set of rectangles (objects) must be positioned on a container (a rectangular space area). This container has a fixed width dimension and a variable height size. The goal is, when possible, to introduce all the objects in the container without overlapping, using a minimum height dimension of the container. In the literature many approaches have been proposed. In our understanding a more complete revision has been presented in E. Hopper's Thesis [10]. However, in the last few years the interest in this subject has increased, as has the interest in the number of research papers presenting new approaches and improvements to the existing strategies. These approaches are in general single heuristics or heuristics incorporated into metaheuristics methods. Recently, the concept of hyperheuristic has been introduced and tested successfully in different problems, [5]. The key idea is to tackle problems using various low-level heuristics and develop a framework that controls the applications of the heuristics. Using this framework the time consuming task of designing an algorithm with special components for a specific algorithm is reduced. This kind of approach is useful to obtain a good solution for a problem in a reasonable amount of time. It emphasises a trade-off between the quality of the solution and the invested time for designing the algorithm.

Our goal in this research is to show that our hyperheuristic can be applied to solve difficult Strip Packing Problems giving good quality solutions in an

* Partially Supported by the Fondecyt Project 1060377.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 406–415, 2007.

efficient way from both points of view: running and designing time. Our approach is compared using well known benchmarks. This paper is organised as follows: First we present an overview of methods based on heuristics to solve the strip packing problem, which are included in our hyperheuristic approach. Next we introduce our framework. We will then present the results obtained using the benchmarks. Finally, our conclusions and future trends in this research area are presented.

2 Heuristics Based Methods

In this section we present a briefly revision of the most recently published heuristics for strip packing problems that are relevant for our approach. Baker introduced [2] Bottom-Left (BL) heuristics, which orders the objects according to their area. The objects are then located on the most bottom and left coordinates possible. BL has been improved by Chazelle [7] using Bottom-Left-Fill (BLF) algorithm. Hopper [11] presented BLD which is an improved strategy of BL, where the objects are ordered using various criteria (height, width, perimeter, area) and the algorithm selects the best result obtained. Lesh et al. in [14] concentrate their research to improve BLD heuristic. They proposed BLD^* where the objects are randomly ordered according to the Kendall-tau distance from all of the possible fixed orders. This strategy is called Bubble Search, [14], the key of the algorithm is the order of the objects to be placed and the object rotation capability. The results reported indicate that the top-right corner is the most suitable decision, and the most effective order is from their minimal length. Bortfeldt [3] introduced a Genetic Algorithm called SPGAL and claimed that it obtained the best results known in the literature. The algorithm generates an initial population using a $BFDH^*$ heuristic which is an improvement on the BFDH heuristic initially proposed in [16]. This heuristic works as follows: The objects are oriented such that their width is no lower than their height, and they are ordered from highest to lowest. Each object is packed in a rectangular sub-area of the container in the bottom left corner. The width of the sub-area is given by the container, and the height is given by the first object packed in this sub-area. In some cases when it is possible to include the current object to be placed on some sub-areas, it is positioned in the sub-area having the least available area. In other cases the algorithm opens a new sub-area above the existing sub-areas positioning the current object in the bottom left corner as the first object of this sub-area. As we mentioned before $BFDH^*$ seeks to improve this heuristic by doing the following: It allows object rotations, so that when the algorithm searches to include the current object into a sub-area it tests both orientations and selects the best. Prior to create a new sub-area the algorithm searches the holes produced to the right of the sub-areas, dividing the available holes on guillotinable holes. It then tries to include the bigger object in the hole farthest left of the available area. Burke et al. [6] proposed the constructive algorithm Best-Fit (BF). The order in which the rectangles are placed into the strip depends on the layout of the partial solution, and the rectangle fitting best into

this layout is selected. Zhang et al. [19] propose the heuristic HR, introducing a recursive algorithm which locates the objects on the bottom left corner. When the first object is positioned in the container it identifies the two remaining areas. It recursively continues placing objects from the biggest area to the lowest area. The algorithm gives priority to the objects with bigger areas. The authors claim that their algorithm quickly obtains the best results on Hopper's benchmarks.

It seems that the key idea is to find a good order of the objects for any positioning heuristic. In [18] they present a genetic algorithm and a simulated annealing algorithm, both of which try to find the best order for the objects to be placed in the container using the BLF strategy. For our hyperheuristics we have selected HR, BF, BLF, $BFDH^*$ as the low-level heuristics, because they are shown to be individually competitive. However, some small adaptations are required for the heuristics designed for guillotinable problems.

3 The Evolutionary Hyperheuristic Approach

From the analysis of the four low-level heuristics we can remark the following:

- Performance changes according to the order of the list of the objects, their rotation, and their location (i.e. right or left on the floor).
- The data structure to obtain a good implementation code is not always the same for all of these heuristics.

Taking into account these remarks we have designed an evolutionary hyperheuristic approach which allows us to include a good individual implementation for each heuristic considering them as black boxes. They communicate following a protocol for both interchanging and cooperation of the current state of the search. Our representation includes the following components: Heuristic H, Number of objects to be placed using H, n_H. The type of ordering of the list of the n_H objects assigned, and finally if H must consider the objects rotated or no.

In this paper we are interested in evaluating a genetic based hyperheuristic which is able to use population capabilities to combine the different heuristics according to each individual fitness. In the following section we describe this approach called G-SP.

3.1 The Genetic Inspired Hyperheuristic: G-SP

Here we propose a new hyperheuristic that is based on genetic algorithms. There exist some genetic inspired hyperheuristics in the literature to solve combinatorial problems, [8], [9]. However, in most of the cases they use a representation that just corresponds to a simple sequence of low-level heuristics to be applied.

Representation. In our approach, we have defined a representation that is able to manage and to exploit more information. We have divided the low-level heuristics according to their functionality. Thus, we distinguish among greedy, ordering

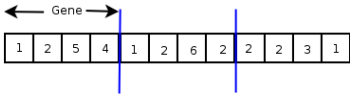


Fig. 1. Chromosome Structure

and rotation heuristics. This kind of representation allows the algorithm to have a wider combination between low-level heuristics. The chromosome has also included the number of objects to be positioned using each low-level heuristics combination. The chromosome structure is shown in figure 1. In this chromosome we can identify that the algorithm must use the first low-level heuristic using the second ordering heuristic applying the fourth rotation heuristic to locate the first five objects. Note that the chromosome has not a fixed size.

Specialised Genetic Operators. The algorithm has four operators. One recombination operator and three mutation-like operators.

- Recombination Operator named Cross-OP: In our approach the recombination operator is an one-point crossover. The cross-point is selected such that a cut inside on the gen single structure is forbidden. It takes two parents to generate two offsprings. After crossing, the operator must do a post-procedure in order to respect the number of objects to be placed by each individual. Either the lacks or the excess of the number of objects are distributed evenly among the genes in the chromosome structure. The goal of this operator, in our approach, is to do exploration of the search space of the low-level heuristics.
- Asexual Operators: Each operator has an especial rôle.
  - Add-OP: The algorithm randomly selects a heuristic H_s from the representation of a selected chromosome. This heuristic has N_s objects to be positioned. A new heuristic is included after H_s. The new heuristic is required to position $n_1 \leq N_s$ of the objects previously assigned to H_s. The n_1 value is randomly selected. The key idea of this operation is to include new heuristics in a different step of the algorithm in order to obtain better cooperation among them.
  - Delete-OP: The algorithm randomly chooses a heuristic from the selected chromosome. The heuristic is then deleted and the number of objects previously assigned, to be located by it, are added to the objects of the previous heuristic. Thus, the algorithm is able to discard some heuristics that are not being relevant to improve its performance.
  - Replace-OP: The algorithm randomly selects both a heuristic to be replaced and the heuristic to be included. The new heuristic included inherits the number of objects to be placed. The other components of its representation are randomly generated. The idea of this operation is to give more exploration capability to the algorithm.

Evaluation Function and Selection. Our approach uses the traditional fitness function for strip-packing [11], that is to minimise the container's height used. It is supposed that the container's width is fixed. A minimisation Roulette Wheel selection is implemented in order to increase the probability of choosing an individual with low height values.

Hyperheuristic Algorithm. Figure 2 shows the hyperheuristic structure. The procedure create_population randomly generate the initial population of individuals. The evaluate_sequence_heuristics applies the low-level heuristics in the order that they appear on the chromosome and it evaluates the chromosome.

```
Pseudocode of G-SP
Begin
iter=0
create_population(pop_size, chromosome_init_size, gene_type)
while max_iter < iter do
   individuals = get_population()
   evaluate_heuristics_sequences(individuals)
   update_individuals_fitness()
   if random < cross_probability
   Cross-OP();
   if random < delete_probability
   Delete-OP();
   if random < add_probability
   Add-OP();
   if random < replace_probability
   Replace-OP();
   update_population()
   iter++
end while
End
```

Fig. 2. Structure of the Hyperheuristic G-SP

3.2 Tuning

The performance of an evolutionary algorithm strongly depends on its parameter values. Because our hyperheuristic is based on an evolutionary approach it experiments the same sensitivity problem. Given that, we have evaluated two approaches to estimate good parameter values for our hyperheuristic. The first one, called ST (i.e., standard tuning), is the classical generate and test procedure. The second approach, called RV, is a new one based on REVAC approach, recently proposed in [17] which uses statistical properties. In the test section we evaluate both schemas.

Tuning Using REVAC. The Relevance Estimation and Value Calibration Approach (REVAC) [17] has been used for tuning. Roughly speaking, REVAC

Table 1. Operator's probabilities tuned using Standard Tuning and REVAC estimations

	ST	RV
Cross-OP	0.3	0.346
Delete-OP	0.33	0.720
Add-OP	0.33	0.323
Replace-OP	0.33	0.282

is a genetic algorithm that uses some statistical properties to determine the better parameter values and also to discard some genetic operators that, with a statistical significance, do not really improve the algorithm to be tuned. It is based on the shannon entropy to measure solutions diversity. The method has shown to be effective, but is a time consuming task because it finds the better values by evaluating many runs of all the problem instances. For this, we have selected the hardest instances that really seem to require the investment in this additional computational effort. The operator's probabilities obtained can be seen in the table 1.

As can we observe, the Standard Tuning (ST) version indicates that the Cross-OP probability is quite less employed than the mutation operators and each asexual operator can be applied with equal probability. For REVAC (RV) we have selected the six hardest instances from the 21 problem instances. The running time for each instance has been fixed in 30 seconds (3 minutes for each instance set). The number of iterations done by REVAC, as it has been recommended by the authors, was 1000 iterations. Thus, the calibration required around 48 hours CPU time. According to the results shown in table 1, we can conclude that the four operators are significant for our hyperheuristic. Note that the higher probability value is for the operator Delete-OP.

4 Tests

We have done two kind of tests. The first one is to compare the results obtained using single low-level heuristics with our hyperheuristic approaches. We report the quality of the solution found and the percentage of each single low-level heuristic used by the hyperheuristic. The second test compares G-SP versions with the better reported results from the state of the art for strip-packing. Both tests use as benchmarks the Hopper's instances [11] for problems $C_1, \ldots, C_7$. The hardware platform for the experiments was a PC Pentium IV Dual Core, 3.4Ghz with 512 MB RAM under the Mandriva 2006 operating system. The algorithm has been implemented in C++.

4.1 Comparison with Low-Level Heuristics

In order to obtain significant results, the hyperheuristic has been executed 10 times for each problem category with various initial populations. We limit the running time to 60 seconds for each problem category.

Table 2. Gap to the solution for: low-level heuristics, hyperheuristic G-SP ST and G-SP RV

	BLF	HR	$BFDH^*$	BF	G-SP ST	G-SP RV
C1	6.6	6.6	6.6	5	0.0	0.0
C2	13.3	8.8	8.8	8.8	4.00	4.00
C3	11.1	6.6	6.6	6.6	3.56	3.33
C4	4.4	3.8	3.8	3.3	1.78	1.67
C5	2.6	2.6	2.6	2.6	1.33	1.22
C6	3.1	2.7	2.7	2.5	1.53	1.56
C7	2.6	2.6	2.6	2.2	1.49	1.65
Average	6.24	4.81	4.81	4.42	1.95	1.91

Gap to the solution: The table 2 shows the percentage from both the optimal solution to the best solution found (gap $\% = \frac{(best_{found} - opt)}{opt}$) and the average for each single heuristic and for the hyperheuristic G-SP with ST and RV tuning.

The quality of the solution found by each single heuristic has been strongly improved using our approaches. Furthermore, the hyperheuristic allows both a division of the task and a cooperation among the heuristics for positioning the objects.

Statistical Comparison: As we mentioned before, our aim is to show that a collaborative schema among simple low-level heuristics improves their individual behaviour. As the low level heuristics considered in this work are deterministic, the gap obtained by applying them at isolated is always the same. However, this is not the case in our hyperheuristics because they could obtain different gaps for the same problem at different runs. Figure 3 shows the boxplots for both hyperheuristics for each problem category. We can observe that the biggest difference among the gaps is obtained in the category C2 for both algorithms. That is because problem 2 in category C2 is very hard for all low-level heuristics

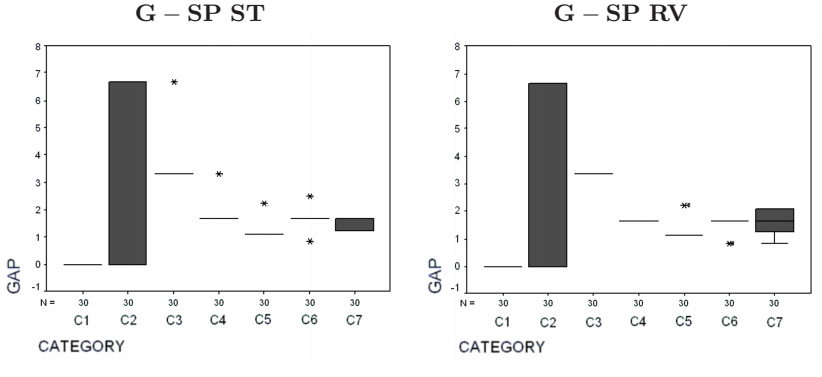


Fig. 3. Boxplots for both G-SP versions

Table 3. Average use of low-level heuristics in the G-SP RV version

	C1	C2	C3	C4	C5	C6	C7	Avg.
BLF	45.43	41.60	50.12	54.49	51.55	58.21	66.91	52.62
HR	15.65	9.87	7.02	1.43	3.93	2.47	4.72	6.44
BFDH*	3.50	28.80	2.74	0.88	1.19	0.55	0.27	5.42
BF	35.42	19.73	40.12	43.20	43.33	38.76	28.10	35.52

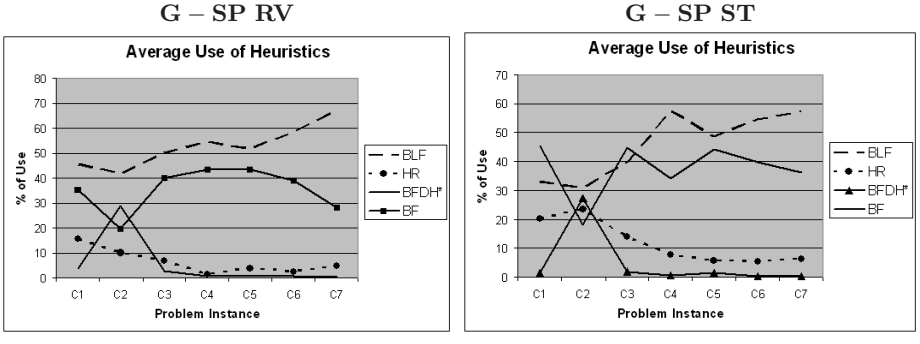


Fig. 4. Percentage of low-level heuristics used for G-SP

as well as for our hyperheuristics. In addition, the hyperheuristic G-SP RV shows more stability than G-SP ST. The above is especially remarkable for problems on categories C3, C4 and C6.

Low-Level Heuristics Runs: In table 3 we report the percentage of the number of times that each heuristic has been applied for each type of problem in our best genetic based hyperheuristic approach for the best heuristics combination.

This table can be interpreted as the number of the objects (in percentage) that each heuristic located on the floor. We can appreciate that each problem requires a different combination of the low-level heuristics. This is the advantage of the implicit natural adaptation of the hyperheuristic framework. A more detailed comparison of the use of the low-level heuristics is shown in figure 4. The figures show that BFDH* tends to be less applied as the size of the problem increases. On the contrary, BLF shows exactly the opposite behaviour. A pattern can not be identified for both BF and HR heuristics. Note however that BF has been used more frequently than HR. In addition, HR is more useful to solve smaller problem categories. Thus, the application percentage of the low-level heuristics depends on the problem instance to be solved. Furthermore, the algorithm is able to self-adapt to the problem at hand.

4.2 Comparison with State-of-the-Art Algorithms

The table 4 summarises the better results found in the literature [11], [12], [19], [3,4], [13], [1], [15], along with the results obtained by our approach for the

Table 4. Gap to the solution for: state-of-the-art algorithms and G-SP tuned versions

	Category							
Technique	C1	C2	C3	C4	C5	C6	C7	Avg.
GA + BLF, [11]	4	7	5	3	4	4	5	4.57
SA + BLF, [11]	4	6	5	3	3	3	4	4
Iori, [12]	1.59	2.08	2.15	4.75	3.92	4.00	-	3.98
HR, [19]	8.33	4.45	6.67	2.22	1.85	2.5	1.8	3.97
SPGAL-R, [4]	1.7	0.0	2.2	0.0	0.0	0.3	0.3	0.6
SPGAL, [3]	1.59	2.08	3.16	2.70	1.46	1.64	1.23	1.98
BLD*, [13]	-	-	-	-	2	2.4	-	2.2
R-GRASP, [1]	0	0	1.08	1.64	1.10	0.83	1.23	0.84
Martello B&B, [15]	0	0	2.15	-	-	-	-	0.71
G-SP ST	0	4.00	3.56	1.78	1.33	1.53	1.49	1.95
G-SP RV	0	4.00	3.33	1.67	1.22	1.56	1.65	1.91

Hopper's instances. Results show that our hyperheuristic versions obtain good quality solutions and even better than various especially-designed algorithms (metaheuristics and heuristics) except for the SPGAL-R and R-GRASP algorithms that present the better solutions. These algorithms have been especially designed for these benchmarks. The above demonstrate that our approach is very competitive. In order to obtain quite good solutions, the parameters must be tuned according to the problem at hand, but this is not the main goal of hyperheuristics. For this reason, we did not invest a large amount of time tuning parameters. Instead of that, we tried to use cheap techniques to adjust the parameters to solve each problem and still giving good quality solutions. In addition, note that the values for HR in this section are not the same of the previous section. In the previous test we have fixed the running time in 60 seconds. Here the results are the best reported for this technique without imposing any time constraint.

5 Conclusions

Our research allows us to conclude that using our evolutionary hyperheuristic approach we can improve the performance of the single heuristics and some of the results obtained in the literature. That indicates that our approach is very promising to solve difficult strip packing problems. The above tell us that the hyperheuristics are strongly rich by having the following characteristics: flexible, cheap and easy to be implemented and, at the same time, are able to obtain quite good solutions. Moreover, our hyperheuristic is able to adapt itself to the problem by selecting the best combination of the low-level heuristics.

We remark that the selection of suitable low-level heuristics is a main task when we design hyperheuristics. In order to obtain competitive solutions with regard to the state of the art, we strongly require to select efficient low-level heuristics. The key idea is to allow the cooperation among them in order to improve their single behaviours.

References

1. Alvarez, R., Parreño, F., Tamarit, J.M.: Reactive grasp for the strip packing problem. In: Proceedings of the 6th Metaheuristics International Conference, vol. 1 (2005)
2. Baker, B.S., Coffman, E.G., Rivest, R.L.: Orthogonal packings in two dimensions. SIAM Journal on Computing 9, 846–855 (1980)
3. Bortfeldt, A.: A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces. European Journal of Operational Research 172, 814–837 (2006)
4. Bortfeldt, A., Gehring, H.: New large benchmark instances for the two-dimensional strip packing problem with rectangular pieces. In: IEEE Proceedings of the 39th Annual Hawaii International Conference on System Sciences, vol. 2, pp. 30–32 (2006)
5. Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., Schulenburg, S.: Hyperheuristics: an emerging direction in modern search technology. Handbook of Metaheuristics 16, 457–474 (2003)
6. Burke, E., Kendall, G., Whitwell, G.: A new placement heuristic for the ortoghonal stock-cutting problem. Operations Research 52, 655–671 (2004)
7. Chazelle, B.: The bottom-left bin packing heuristic: an efficient implementation. IEEE Transactions on Computers 32, 697–707 (1983)
8. Cowling, P., Kendall, G., Han, L.: An investigation of a hyperheuristic genetic algorithm applied to a trainer scheduling problem. In: Proceedings of Congress on Evolutionary Computation, pp. 1185–1190 (2002)
9. Han, L., Kendall, G.: Guided operators for a hyper-heuristic genetic algorithm. In: Gedeon, T.D., Fung, L.C.C. (eds.) AI 2003. LNCS (LNAI), vol. 2903, pp. 807–820. Springer, Heidelberg (2003)
10. Hopper, E.: Two-Dimensional Packing Utilising Evolutionary Algorithms and other Meta-Heuristic Methods. PhD. Thesis Cardiff University, UK (2000)
11. Hopper, E., Turton, B.C.H.: An empirical investigation on metaheuristic and heuristic algorithms for a 2d packing problem. European Journal of Operational Research 128, 34–57 (2001)
12. Iori, M., Martello, S., Monaci, M.: Metaheuristic algorithms for the strip packing problem, pp. 159–179. Kluwer Academic Publishers, Dordrecht (2003)
13. Lesh, N., Marks, J., Mc Mahon, A., Mitzenmacher, M.: Exhaustive approaches to 2d rectangular perfect packings. Information Processing Letters 90, 7–14 (2004)
14. Lesh, N., Mitzenmacher, M.: Bubble search: A simple heuristic for improving priority-based greedy algorithms. Information Processing Letters 97, 161–169 (2006)
15. Martello, S., Monaci, M., Vigo, D.: An exact approach to the strip-packing problem. INFORMS Journal of Computing 15, 310–319 (2003)
16. Mumford-Valenzuela, C., Vick, J., Wang, P.Y.: Heuristics for large strip packing problems with guillotine patterns: An empirical study, pp. 501–522. Kluwer Academic Publishers, Dordrecht (2004)
17. Nannen, V., Eiben, A.E.: Relevance estimation and value calibration of evolutionary algorithm parameters. In: International Joint Conference on Artificial Intelligence, pp. 975–980 (2007)
18. Soke, A., Bingul, Z.: Hybrid genetic algorithm and simulated annealing for two-dimensional non-guillotine rectangular packing problems. Engineering Applications of Artificial Intelligence 19, 557–567 (2006)
19. Zhang, D., Kang, Y., Deng, A.: A new heuristic recursive algorithm for the strip rectangular packing problem. Computers and Operations Research 33, 2209–2217 (2006)

Statistical Analysis of Sample-Size Effects in ICA

J. Michael Herrmann[1,2,⋆] and Fabian J. Theis[1,3,⋆⋆]

[1] Bernstein Center for Computational Neuroscience Göttingen
[2] Göttingen University, Institute for Nonlinear Dynamics
michael@chaos.gwdg.de
[3] Max Planck Institute for Dynamics and Self-Organization
Bunsenstraße 10, 37073 Göttingen, Germany
fabian@theis.name

Abstract. Independent component analysis (ICA) solves the blind source separation problem by evaluating higher-order statistics, e.g. by estimating fourth-order moments. While estimation errors of the kurtosis can be shown to asymptotically decay with sample size according to a square-root law, they are subject to two further effects for finite samples. Firstly, errors in the estimation of kurtosis increase with the deviation from Gaussianity. Secondly, errors in kurtosis-based ICA algorithms increase when approaching the Gaussian case. These considerations allow us to derive a strict lower bound for the sample size to achieve a given separation quality, which we study analytically for a specific family of distributions and a particular algorithm (fastICA). We further provide results from simulations that support the relevance of the analytical results.

1 Introduction

Independent component analysis (ICA) aims at the extraction of source signals from measured multi-dimensional data [10, 5]. In order to identify a transformation of the data such that the resulting random vector is statistically independent, probabilistic algorithms for ICA rely on estimates of higher moments of the probability distribution functions of the observations [8]. While for first- and second-order moments many mathematical results (cf. e.g. [12]) are available describing the dependence of the estimation quality on the sample size or on the underlying parameters, the statistical properties of estimates of higher moments are less understood. A useful but typically loose characterization of the sample-size effects is given by the Tschebyscheff inequality, which assures a square-root decay of large deviations, provided that the existence of certain moments is guaranteed. Tighter estimates of higher moments are often designed as tests of normality, i.e. work best in the vicinity of the normal distribution. Probabilistic ICA algorithms are, however, based on the assumption that all but at most one sources are non-Gaussian [5, 15]. Thus on the one hand, ICA algorithms prefer source distributions most of which are sufficiently far from Gaussianity, while on the other hand estimates of higher-order moments such as the kurtosis are increasingly inefficient on deviations from Gaussianity [3].

⋆ Present address: University of Edinburgh, Div. of Informatics, Edinburgh, Scotland, UK.
⋆⋆ Present address: Computational Modeling in Biology, IBI, GSF, Munich, Germany.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 416–425, 2007.

Theoretical results in independent component analysis commonly focus on either showing existence and uniqueness of solutions [5,15] or derive conditions that lead to considerable improvements in separation quality. For example, the learning rate can be bounded such as to achieve monotonous convergence [14] or can be modulated in order saturate the Cramér-Rao bound of the variance of the off-diagonal elements of the confusion matrix [11]. These conditions are formulated, however, in terms of certain expectation values of the contrast function, while the estimation errors for these quantities tend to be ignored such that essentially only asymptotic results are obtained. In the case of real-world data sets both the moments of the contrast function are unknown and the set of samples is small [6]. A serious study of data from natural images [4] reports a negligible gain in statistical independence by ICA when compared to a decorrelation of the set of image patches by principal component analysis. Reasons for this failure may be found in the questionable validity of the assumptions underlying the ICA algorithm.

The present study identifies several causes of ICA-based component estimation errors, namely sample-size, the statistical properties of the sources such as the distribution of kurtoses, and temporal correlations. We restrict the analysis to a single popular algorithm, fastICA [9], and to a specific family of distributions in order to obtain analytical results. The obtained results can be understood as counterexamples to assumptions which are also made in a larger class of algorithms and for other data distributions. We have chosen Pearson's second and seventh family [1,7], see section 3.2 and Fig. 1. The choice of this example is suggestive: Zero mean and unit variance of the observations can be assumed without much loss of generality, since centering and whitening the data are non-critical in most cases. If additionally symmetry is assumed, then the kurtosis κ is the most relevant characteristic of the distribution function. κ is defined by the ratio of the fourth central moment $\mu_4 = E[(x-\mu)^4]$ and the squared variance σ^4, i.e. for a normal (Gaussian) distribution we have $\kappa = 3$. After appropriate reparameterization, Pearson's families depend on κ as the single parameter. The main advantage we are relying on here consists in the controlled access to a wider range of kurtoses than typically found in real data, where many components may be close to Gaussianity.

We start with a brief review of the blind source separation paradigm and discuss the role of estimates of moments in fastICA [9,8]. Section 3 then presents analytical results concerning the optimally achievable estimation quality in the Cramér-Rao sense, and compares this bound with the expected performance of a particular estimation scheme. In section 4, we illustrate these results by a set of simulations, which demonstrate that sample-size effects are drastic for some parameter settings. Finally, in Section 5, we conclude by discussing implications of the presented results for practical applications of ICA algorithms.

2 Blind Source Separation

The linear blind source separation model describes the mixture of signals by a matrix multiplication

$$\mathbf{X} = \mathbf{AS} \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^d$ represents the observed data and $\mathbf{S} \in \mathbb{R}^n$ the underlying source signals, both modeled by random vectors. A full-rank matrix $\mathbf{W} \in \mathbb{R}^{n \times d}$ is said to be a solution

of the ICA of $\mathbf{X}$ if $\mathbf{WX}$ is statistically independent. We assume that $n = d$ and that $\mathbf{S}$ has at most one Gaussian component. It can be shown that $\mathbf{W}^{-1}$ equals $\mathbf{A}$ except for right-multiplication by a permutation and a scaling matrix [5,15].

We are interested in how many samples are needed to guarantee certain recovery rate with a confidence of say $\alpha = 95\%$. In order to study this question, we make the following restrictions in addition to the linearity and independence assumption:

- We only consider higher-order statistics, i.e. assume that the data have been whitened and $\mathrm{Cov}(\mathbf{X}) = \mathrm{Cov}(\mathbf{S}) = \mathbf{I}$, without estimation errors. This allows to consider only orthogonal $\mathbf{A}$ and $\mathbf{W}$. Further, we assume vanishing source skewnesses (such as in the case of sources with symmetric densities) and non-trivial kurtoses, so that we can restrict ourselves to fourth-order statistics.
- We restrict ourselves to 'deflation', i.e. the one-unit approach, where a single column of the mixing matrix is extracted, such as in the fastICA algorithm [9,8].
- In order to reduce the semi-parametric model to a well-defined estimation problem, we consider sources from Pearson's family, cf. Fig. 1.

Our goal is to invert (1), i.e. to find $\mathbf{W}$ with $\mathbf{WX}$ independent. In deflation mode, this is done one-by-one by only extracting a single source within $\mathbf{X}$ i.e. a single row $\mathbf{w}_i^\top$ of $\mathbf{W}$. It can be shown that this can be achieved by maximizing the kurtosis $\mathrm{kurt}(\mathbf{w}_i^\top \mathbf{X})$ of the transformation [9,8], where the (normalized) kurtosis $\mathrm{kurt}(S) := \kappa(S) - 3$ is zero if S is normal.

Starting with a minimum-mutual-information approach, we get $I(\mathbf{WX}) = \sum_i H(S_i) - H(\mathbf{X})$ (assuming orthogonal $\mathbf{W}$), with $H(\mathbf{X})$ being constant. This equals maximizing the above general likelihood because $\sum_t \log p(\mathbf{w}_i^\top \mathbf{x}(t)) = E(\log p(\mathbf{w}_i^\top \mathbf{X})) = H(\mathbf{w}_i^\top \mathbf{X})$. So $I(\mathbf{WX}) = c - \sum_i J(S_i)$ with some constant c independent of $\mathbf{W}$ and the negentropy $J(S_i) = H(S_{i,\mathrm{gauss}}) - H(S_i)$. Using Gram-Charlier expansion of densities, the negentropy can be approximated as

$$J(S_i) = \frac{1}{12} E(S_i^3)^2 + \frac{1}{48} \mathrm{kurt}(S_i)^2 + \text{higher-order statistics}. \tag{2}$$

Due to our assumption of vanishing skewness we may approximate $\sum_i J(S_i)$ by

$$\sum\nolimits_i \mathrm{kurt}(\mathbf{w}_i^\top \mathbf{X})^2. \tag{3}$$

Instead of (3) we consider now the related maximization problem $\sum_i \sigma_i \mathrm{kurt}(\mathbf{w}_i^\top \mathbf{X})$ with $\sigma_i := \mathrm{sign}\,\mathrm{kurt}(\mathbf{w}_i^\top \mathbf{X})$. Relying on the independence of the sources we can write

$$\mathrm{kurt}(\mathbf{w}^\top \mathbf{X}) = \mathrm{kurt}(\mathbf{v}^\top \mathbf{S}) = \sum\nolimits_i v_i^4 \,\mathrm{kurt}(S_i), \tag{4}$$

where $\mathbf{v}^\top := \mathbf{w}^\top \mathbf{A}$ with $\|\mathbf{v}\| = 1$ [9]. The extrema of the expression (4) are assumed for $\mathbf{v} = \pm \mathbf{e}_i$ for some i, where $\mathbf{e}_i$ denotes the i-th unit vector. But since $\mathbf{A}$ is unknown, $\mathbf{v}$ is an implicit variable. Therefore the result of the ICA i.e. the extraction of the vector $\mathbf{w}$ strongly depends on the quality of the estimation of the kurtosis.

The performance of the ICA algorithm depends thus essentially on the error of the empirical kurtosis $\widehat{\mathrm{kurt}}(\mathbf{w}^\top \mathbf{X})$. Denoting the random variable $\mathrm{kurt}(\mathbf{w}^\top \mathbf{X}) - \widehat{\mathrm{kurt}}(\mathbf{w}^\top \mathbf{X})$ by Δ, we realize that, instead of the ideal result $\mathbf{v} = \pm \mathbf{e}_i$, Eq. (4) yields.

$$\left(v_j \mp e_{ij}\right)^4 \sim \frac{\Delta}{n\,\mathrm{kurt}(S_j)}, \tag{5}$$

where e_{ij} is the jth component of the unit vector $\mathbf{e}_i$. Typically, Δ is of the order of the variance of $\widehat{\mathrm{kurt}}(\mathbf{w}^\top \mathbf{X})$, below we will analyze the latter random variable.

3 Finite-Sample Effects in the Estimation

We will proceed in three steps. First we consider the quality of the commonly used estimator for the kurtosis $\hat{\kappa} = \hat{\mu}_4$. It is unbiased since sample-size dependent weighting is not required when the lower moments are assumed to be known. Next we discuss the minimal variance of an unbiased estimator of the kurtosis based on the Cramér-Rao inequality, cf. 3.3. Finally we will present results from simulations in section 4, emphasizing especially scaling effects in the sample size, whereas the first two analytical consideration concentrate on the effect *per* sample.

3.1 A Test for Kurtosis

Kurtosis is a relevant property in tests of normality of a empirically given distribution. In this context it is natural to assume kurtosis to be small. Estimators that are efficient far from Gaussianity are not known, and the following theorem provides some insight why this is the case:

Theorem [3]: If the kurtosis κ is estimated by the sample moments $\hat{\mu}_r = N^{-1}\sum_{n=1}^{N}(x_n-\mu)^r = E(x_n-\mu)^r$, the resulting estimator $\hat{\kappa}$ deviates from the true kurtosis as follows:

$$(\hat{\kappa}-\kappa) = \frac{\beta}{\hat{\sigma}^4}\frac{1}{N}\sum\nolimits_{n=1}^{N}\mathbf{B}_n + o\left(1\right), \tag{6}$$

where $\beta = \left(1, -4\mu_3, -2\sigma^2\kappa\right)$ and $\mathbf{B}_n = \left((x_n-\mu)^4 - \mu_4, x_n-\mu, (x_n-\mu)^2-\sigma^2\right)^\top$. The theorem refers to the estimator of kurtosis that is given by the estimates of the fourth central moment and the variance. Because we have fixed the variance to unity, we can write $\hat{\sigma}^2 = \sigma^2 = 1$ and the kurtosis estimate is the only source of the error. The estimator of the kurtosis is asymptotically normal [3] and its variance is given by $\hat{\sigma}^2_\kappa = \hat{\beta}^\top E[\hat{\mathbf{B}}\hat{\mathbf{B}}^\top]\hat{\beta}/N$. In our case of a non-skewed distribution with unit variance this expression simplifies to

$$E[(\kappa - 2\kappa x^2 + x^4)^2]/N. \tag{7}$$

This result will be used below as an approximation of Δ in Eq. 5.

3.2 Pearson's Families

The family of leptokurtotic (super-Gaussian) distribution functions defined by

$$f_{\mathrm{VII}}\left(x;m\right) = \frac{\left(1+\frac{\kappa_0 x^2}{6+2\kappa_0}\right)^{-m}\Gamma\left(m\right)}{\sqrt{1+\frac{3}{\kappa_0}}\sqrt{2\pi}\Gamma\left(m-\frac{1}{2}\right)} \tag{8}$$

is known as Pearson type VII [1]. The distributions are symmetric. For $m = \frac{5}{2} + \frac{3}{\kappa_0}$ with $\kappa_0 = \kappa - 3$ the distribution is singly-parameterized by its kurtosis $\kappa > 3$.

The platykurtotic Pearson type II family is given by

$$f_{\mathrm{II}}(x; r) = \frac{\Gamma\left(r + \frac{3}{2}\right)}{\Gamma(r+1)} \frac{1}{\sqrt{\pi r^2}} \left(1 - \frac{(x-\mu)^2}{r^2}\right)^r . \tag{9}$$

f_{II} is symmetric for $\mu = 0$. With the further substitutions $r = \sqrt{3 + 2m}$ and $m = \frac{3}{3-\kappa} - \frac{5}{2}$, it is parameterized by its kurtosis $\kappa \in (1, 3)$. The Gaussian case yields in both families as the limit $\kappa \to 3$ such that we can consider the two families as connected.

For the platykurtotic family the variance decays with the distance from the Gaussian case, cf. Fig. 2(a). This indicates that a reliable estimate of the kurtosis can be obtained already at relatively small sample sizes. Intuitively, this is due to the dramatic structural changes that the distributions of the platykurtotic family undergo when varying κ. Thus a few samples are sufficient to identify a confidence interval for the kurtosis. For $\kappa \to \infty$, however, the density functions of the leptokurtotic family slowly approach the limiting distribution $3(2 + x^2)^{-5/2}$, which does not differ too much from the Gaussian distribution as can be seen in Fig. 1. The variance of the kurtosis increases (cf. Fig. 2(b)) because the changes of the form of the density become smaller and smaller. Note that the absolute values in Fig. 2 refer to the variance of the estimator from a single sample and are to be divided by the sample size.

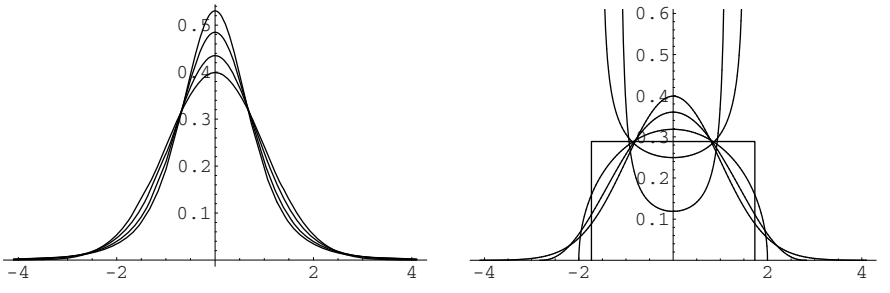


Fig. 1. The left subfigure shows the leptokurtotic (or super-Gaussian) Pearson VII family, which equals the normalized Student-t distribution. All distribution functions (given here for the values $\kappa = 3,\ 4,\ 8$, and $\kappa \to \infty$) are uni-modal and vary slowly in shape for increasing κ. For $\kappa \to \infty$ the family approaches the limit distribution $p(x) = 3(2 + x^2)^{-5/2}$, $\kappa \to \infty$, where strictly speaking the kurtosis does not exist. For $\kappa = 3$, we get the standard normal distribution. The right figure represents the platykurtotic (or sub-Gaussian) Pearson II family. The curve with the largest central peak is again the Gaussian, which connects both families at $\kappa = 3$. Further values of κ are 2.4, 2, 1.8, 1.6 and 1.2. At $\kappa = 2$ the distribution function forms a semi-ellipse, while for $\kappa = 1.8$ it is of box-shape, i.e. the uniform distribution is a member of this family. Eventually, at $\kappa = 1$ the function degenerates into a sum of two δ-distributions which are located at $x = \pm 1$ (not shown). For $1 \leq \kappa < 3$ the support of the functions is bounded by $\pm\sqrt{\frac{6}{3-\kappa} - 2}$.

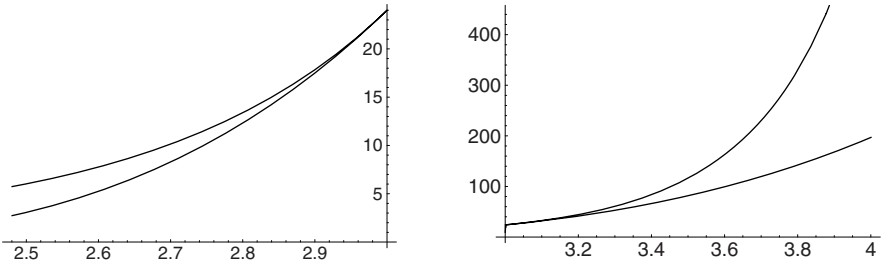


Fig. 2. Cramér-Rao bound (lower curves) for the estimation of the kurtosis κ as the parameter of the platykurtotic Pearson family (left) and the leptokurtotic Pearson family (right). The upper curves represent the respective variances of the kurtosis estimator [3]. The variance of the estimator coincides with the inverse Fisher information and is thus efficient only in the Gaussian limit ($\kappa = 3$), where we find consistently $J^{-1}(3) = 24$.

3.3 Fisher Information

The Cramér-Rao inequality $\mathrm{Var}(\hat{\kappa}) \geq J(\kappa)^{-1}$ states that the variance of any specific unbiased estimator for the kurtosis is bounded from below by the inverse of the Fisher information

$$J(\kappa) = \int p(x;\kappa) \left(\frac{\partial \log p(x;\kappa)}{\partial \kappa} \right)^2 dx, \tag{10}$$

where the integral is taken over the support of the density function $p(x)$. The integral (10) can be calculated analytically for Pearson's families in the sense of a principal value. The respective formulas have been obtained using Mathematica and the result is graphically displayed in Fig. 2.

An unbiased estimator is efficient if it meets the Cramér-Rao bound. The above estimator of the kurtosis [3] achieves minimal variance only in the Gaussian case. For non-trivial kurtoses the quality of the estimator deviates considerably, see Fig. 2.

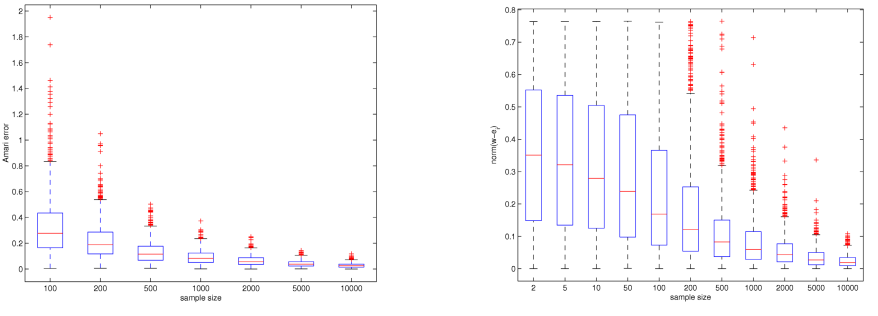


Fig. 3. Cross-talking error $E_1(\mathbf{WA})$ (11) for total matrix (left) and recovery error (12) for a single column (right). The boxes are delimited by the quartiles and intersected by the median. Data are marked by crosses if their distance from the box is more than 1.5 times the interquartile distance.

Table 1. T_0 is defined as the maximal number of samples where still the maximum error is likely to be observed, while T_{min} is the minimal number of patters which is needed in order to achieve an average error (12) of 0.1. And $T_{\alpha=5\%}$ is the error (0.039) at the 5%-quantile, when sampling random directions and calculating the resulting error.

κ	1.5	1.6	1.8	2.0	2.5	2.9	3.0	3.1	4.0	6.0	9.0	13.0
T_0	3	5	9	17	146	4242	∞	6031	184	80	65	58
T_{min}	71	96	189	319	2251	72829	∞	85464	2557	1026	869	723
$T_{\alpha=5\%}$	457	617	1218	2060	14562	471536	∞	553346	16542	6637	5620	4675

4 Simulations

At first, we study the estimation error by considering a $(n = 2)$-dimensional uniform distribution with covariance $\mathrm{Cov}(\mathbf{S}) = \mathbf{I}$. The differences of the recovered demixing matrix $\mathbf{W}$ from the mixing matrix $\mathbf{A}$ can be evaluated by Amari's performance index

$$E_1(\mathbf{C}) = \sum_{r=1}^{n} \sum_{s=1}^{n} \left(\frac{|c_{rs}|}{\max_i |c_{ri}|} + \frac{|c_{sr}|}{\max_i |c_{ir}|} - 2 \right), \tag{11}$$

which quantifies deviations of $\mathbf{C} = \mathbf{WA}$ from a scaling and permutation matrix [2]. This measures the performance of the full ICA algorithm. For analyzing a single deflation step we make use of the single column error

$$E_2(\mathbf{v}) := \min_{\mathbf{e}_i} \|\mathbf{v} \pm \mathbf{e}_i\|. \tag{12}$$

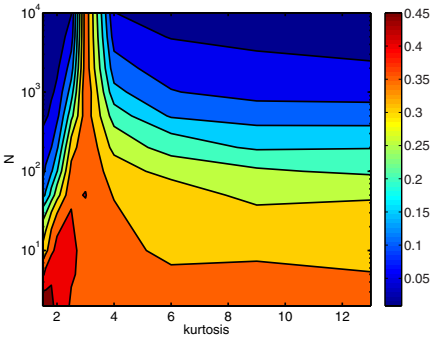
The comparison is made over 1000 runs, in which the coefficients of $\mathbf{A}$ have been drawn uniformly from the orthogonal group. In Fig. 3, we see that for varying number of samples N, indeed the error gets smaller, roughly following the expected $1/\sqrt{N}$-law.

Now, we calculate the mean and standard deviation when varying the number of samples N and the source kurtosis κ (assumed to be equal for all source components). This is done in $n = 2$ dimensions for 1000 batch runs to get sufficient statistics. In Fig. 4, the results are presented. Clearly the error increasing to randomness in the case of Gaussians ($\kappa = 3$) as was to be expected due to the symmetry in that case. For values different from 3, recovery error versus N can be analyzed, and we observe that this relationship follows a power-law with exponent $-\frac{1}{2}$, confirming again the asymptotic $1/\sqrt{N}$-decrease, cf. Eq. 7.

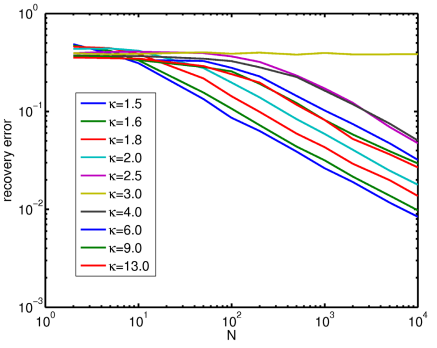
A further set of simulations studies the effect of temporal correlations which is included here because it shows a behavior opposite to the intuition implied by Fig. 2. For the more structured platykurtotic distributions, temporal correlations yield a larger recovery error than the less informative leptokurtotic distributions. The obtained differences are significant for the illustrative case studied here and cannot be ignored in less obvious situations.

5 Discussion

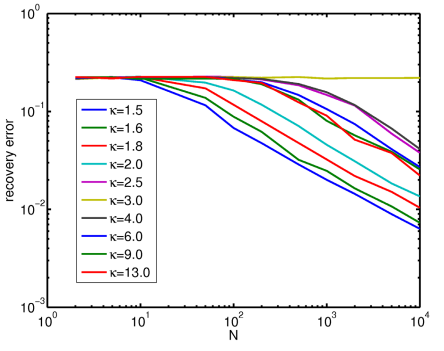
For large sample sizes the square-root decay of both the error and the variance reproduces relations (6) and (7), In Figs. 4b and c it is obvious that the intercept of the



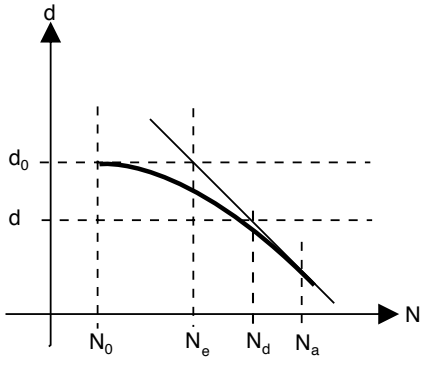
(a) Mean matrix-row estimation error



(b) Mean recovery error versus number of samples N for varying kurtosis



(c) Standard deviation of the recovery error versus number of samples N



(d) Construction of maximal N bounds

Fig. 4. Comparison of row estimation of $\mathbf{W}$ in the symmetric Pearson family for varying number of samples N and kurtosis κ. Mean and variance are taken over 1000 runs. Figures (a) and (c) show the mean, and (b) the corresponding standard deviation. The asymptotic $1/\sqrt{N}$-decrease can be used to fit a line in the log-log plots and to determine its intersection with the maximal error to get bounds for minimal number of samples N_e and N_d depending on the error threshold.

log-linear fit to the asymptotic part of the equation depends clearly on the true value of the kurtosis that is estimated. For sample sizes below the intercept, no information about the component is extracted from the mixture. The respective values are corrected for the estimation quality as discussed in section 3.3 and are illustrated by the values in Tab. 1. They can be obtained from the simulation as illustrated in Fig. 4 d. Asymptotically we can assume $1/\sqrt{N}$-decay, say at $N = N_a$. We are then interested at the number of samples N_d that produce precisely an error of $d(N_d)$ according to this upper bound. This can be easily derived as $N_d = N_a d_a^2/d^2$. Interesting upper bounds N are N_e for the maximal error $d_{\max}$, which is in practice achieved at much smaller samples, and the maximal error at confidence level of α, where statistics are performed over randomly sample distances and resulting errors d, see Tab. 1.

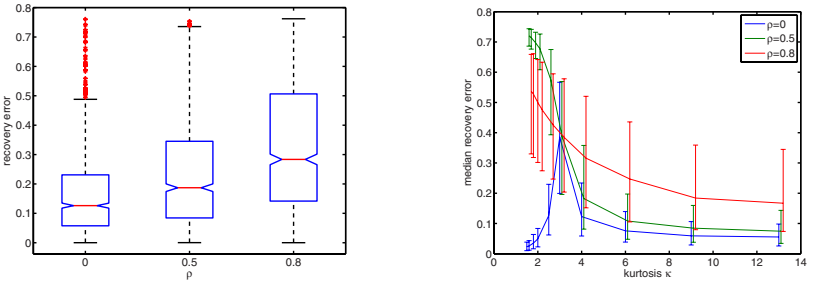


Fig. 5. $T = 1000$ samples of a two-dimensional problem are drawn from a Pearson distribution with kurtosis $\kappa = 4$ with a temporal correlation $X(t) = \rho X(t-1) + \epsilon(t)$, where $\epsilon(t)$ are i.i.d. random variables. Presented is an average over 1000 runs for each value ρ. In the plot on the left, the depths of the dents next to the median (compare Fig. 3) indicate that the significancy of the difference according to the Wilcoxon rank sum test ($\rho = 0.0$ vs. $\rho = 0.5$: $p = 3.1 \times 10^{-15}$, $\rho = 0.0$ vs. $\rho = 0.8$: $p = 3.5 \times 10^{-62}$, and $\rho = 0.5$ vs. $\rho = 0.8$: $p = 4.1 \times 10^{-21}$). The plot on the right shows the median and the quartiles of the recovery error in dependency of the source kurtosis κ for different levels of temporal coherency.

Combining Eqs. 12 and 5 we notice that kurtoses $\kappa \approx 3$ will have a larger effect on the ICA errors than inefficiencies of the estimator. For the estimate of the minimal sample size we can rely on the additivity of the Fisher information provided that samples are independently chosen. In this case we can derive a heuristic correction from the result in Fig. 2. We multiply the minimal sample size by the square root ratio of the variances of the used estimator [3] and the Cramér-Rao bound, which partially compensates the dependency of the minimal sample size on the true kurtosis. For temporally correlated samples, however, the errors are more grave, cf. the following section and Fig. 5.

6 Conclusion

In order to arrive at a practically useful criterion to determine the minimal sample size in ICA, we have chosen Pearson's family as a prototypical case, because it is non-trivially focused on the problem of kurtosis estimation while analytical consideration are still possible. Based on the presented results we claim that often, e.g. in the analysis of fMRI data sample sizes are too small to permit interpretable results. This situation often becomes more critical because of further sources of errors. In kurtotic distributions, deviations from the mean are either the common (platykurtotic case) or drastic because of heavy tails (leptokurtotic case) and the presence (or absence) of extreme values of the observation has a large effect on the estimate of any higher-order moments. This is also indicated by the local scores of the Fisher information tend to be large at these extreme values, such that outlier rejection is a questionable option. On the other hand the study of temporally correlated samples, cf. Fig. 5 shows that less informative families of distributions are less vulnerable to this type of deviations from the standard assumptions of ICA.

The overall picture that emerges here can be summarized as follows: While source distributions of near Gaussianity are naturally difficult, both, lepto- and platykurtotic distributions may cause substantial errors in the reconstructed source signals which persists up to moderate sample sizes. The results obtained here are likely to be characteristic for larger classes of probability distributions, although also other factors may then affect the errors. In addition to the errors due to finite-sample effects, obviously also nonstationarities or violations of the independence assumption may influence the result. Algorithms [13] that exploit temporal correlations require less severe restriction, and may thus avoid some of the problems discussed here. Finally we want to refer the reader to effects of finite sample sizes in the estimation of lower-order moments, for which exist more studies [12] than for the higher-order estimation problem.

Acknowledgment. This work was partially supported by the BMBF, grant number 01GQ0432. Discussions with T. Geisel are gratefully acknowledged.

References

1. Abramowitz, M., Stegun, I.A. (eds.): Handbook of mathematical functions with formulas, graphs, and mathematical tables. Dover, Mineola (1972)
2. Amari, S., Cichocki, A., Yang, H.H.: A new learning algorithm for blind signal separation. Advances in Neural Information Processing Systems 8, 757–763 (1996)
3. Bai, J., Ng, S.: Tests for skewness, kurtosis, and normality for time series data. J. Business & Economic Statistics 23(1), 49–60 (2005)
4. Bethge, M.: Factorial coding of natural images: how effective are linear models in removing higher-order dependencies? J. Opt. Soc. Am. 23(6), 1253–1268 (2006)
5. Comon, P.: Independent component analysis - a new concept? Signal Processing 36, 287–314 (1994)
6. Dodel, S., Herrmann, J.M., Geisel, T.: Comparison of temporal and spatial ica in fmri data analysis. In: ICA 2000. Proc. Second Int. Workshop on Independent Component Analysis and Blind Signal Separation, vol. 13, pp. 543–547 (2000)
7. Farvardin, N., Modestino, J.: Optimum quantizer performance for a class of non-gaussian memoryless sources. IEEE Trans. Inf. Theory IT-30, 485–497 (1984)
8. Hyvärinen, A., Karhunen, J., Oja, E.: Independent component analysis. John Wiley & Sons, Chichester (2001)
9. Hyvärinen, A., Oja, E.: A fast fixed-point algorithm for independent component analysis. Neural Computation 9, 1483–1492 (1997)
10. Jutten, C., Hérault, J., Comon, P., Sorouchiary, E.: Blind separation of sources, parts I, II and III. Signal Processing 24, 1–29 (1991)
11. Koldovský, Z., Tichavský, P., Oja, E.: Efficient variant of algorithm fastICA for independent component analysis attaining the Cramér-Rao lower bound. IEEE Transactions on Neural Networks 17(5), 1265–1277 (2006)
12. MacCallum, R.C., Browne, M.W., Sugawara, H.M.: Power analysis and determination of sample size for covariance structure models. Psychological Methods 1(2), 130–149 (1996)
13. Molgedey, L., Schuster, G.: Separation of a mixture of independent signals using time delayed correlations. Physical Review Letters 72(23), 3634–3637 (1994)
14. Regalia, P.A., Kofidis, E.: Monotonic convergence of fixed-point algorithms for ICA. IEEE Transactions on Neural Networks 14(4), 943–949 (2003)
15. Theis, F.J.: A new concept for separability problems in blind source separation. Neural Computation 16, 1827–1850 (2004)

HPGP: An Abstraction-Based Framework for Decision-Theoretic Planning

Letícia Friske and Carlos Henrique Costa Ribeiro

Divisão de Ciência da Computação - Instituto Tecnológico de Aeronáutica
12228-900 São José dos Campos – Brasil
{leticia, carlos}@ita.br

Abstract. This paper is a report on research towards the development of an abstraction-based framework for decision-theoretic planning. We make use of two planning approaches in the context of probabilistic planning: planning by abstraction and planning graphs. To create abstraction hierarchies our planner uses an adapted version of a hierarchical planner under uncertainty, and to search for plans, we propose a probabilistic planning algorithm based on Pgraphplan. The article outlines the main framework characteristics, and presents results on some problems found in the literature. Our preliminary results suggest that our planner can reduce the size of the search space, when compared with Pgraphplan, hierarchical planning under uncertainty and top-down dynamic programming.

Keywords: Probabilistic Planning, PGraphplan and Dynamic Programming.

1 Introduction

Robust plan creation in probabilistic domains is a complex problem that can use a variety of algorithms and techniques. Recent research in probabilistic planning is focused on MDPs [1] and Dynamic Programming [2] or in state-space search methods. An interesting proposal describes probabilistic problems on a Probabilistic Planning Domain Definition Language - PPDDL [3], so that can it be solved as a STRIPS planning problem. One limitation in this sense is that most existing STRIPS planners are for deterministic domains, forcing these solutions to be adapted for operation under uncertainties.

Some planners, like Buridan [4], Weaver [5], C-SHOP [6], Drips [7], PGraphPlan [8, 9] and TGraphPlan [8], Paragraph [10] and Prottle[11] propose extensions to STRIPS classical planners, to handle situations where action effects are probabilistic.

The Buridan planner uses partial-order planning to build plans that probably achieve a goal. Weaver is a planner that can handle uncertainty about actions taken by external agents, more than one possible initial state and non-deterministic outcomes of actions. It produces conditional plans and computes the plan's probability of success automatically, through a Bayesian belief net. C-SHOP extends the classical hierarchical planner SHOP [12] to act in situations with incomplete and uncertain information about the environment and actions. Drips combines conditional planning

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 426–436, 2007.

with probabilistic distribution of effects, and accomplishes abstraction of probabilistic operators with common pre-conditions and effects. This mechanism makes it possible that planning happens in different hierarchical abstraction levels. Pgraphplan and Tgraphplan are extensions of the classical planner Graphplan to probabilistic planning. Paragraph and Prottle extends the Graphplan framework for concurrent probabilistic planning. They use the planning graph primarily for computing heuristic estimates for a forward search.

Unfortunately, the inclusion of probabilistic effects causes an explosive growth in the state-space search and dynamic programming methods. This leads us to investigate approaches that efficiently deal with probabilistic operators and large problems, which is the focus of this work. We consider two strategies to reduce the search: planning by abstraction [13] (e.g. ABSTRIPS [14], Alpine [15], HW [16]) and planning graphs [17], because both demonstrate significant reductions in the search space when compared with other search strategies.

We then propose a hierarchical probabilistic planner, named HPGP – Hierarchical Probabilistic Graphplan. HPGP automatically builds its abstraction hierarchy and executes the hierarchical control planning based on the hierarchical planner HIPU [18]. The search is based on an adaptation of PGraphplan [8].

In this article, we outline the main characteristics of the abstraction-based framework for decision-theoretic planning, and present case studies in the Blocks World problem and in the Flat-Tire domain, that are classical examples for autonomous manipulation planning. Manipulation planning is concerned with handling objects, e.g., to build assemblies. Actions include sensory-motor primitives that involve forces, touch, vision, range detection, and other sensory information. A plan might involve picking up an object from its marked sides, returning it if needed, inserting it into an assembly, etc [19]. The remainder of this paper is organized as follow. Section 2 presents basic definitions on planning and the notation that will be adopted along the article. Section 3 provides some introductory knowledge about Graphplan and Pgraphplan. Section 4 presents the automatic generation of abstraction hierarchies in HIPU and demonstrates the resolution of hierarchical problems. Section 5 discusses the HPGP framework. Empirical results are presented in Section 6. Finally, Section 7 proposes future work and concludes this paper.

2 Planning in Uncertain Environments

One of the common uncertainties that an agent can find regards the actions, which can have stochastic effects. In other words, the next environmental state is expressed as a probability distribution over states. Planning problems with this kind of uncertainties have been approached in Artificial Intelligence through adaptation of problems of classical planning and high-level languages, such as STRIPS, to act in uncertainty domains.

The adaptations proposed here modify the definition of the classical planning problem, specifically regarding the operators. The space problem is defined by a set of probabilistic operators, where each operator consists of preconditions and one or more subsets of effects. For each subset of effects there is an associated occurrence probability $\wp_i(E^i_\alpha)$, such that $0<\wp_i(E^i_\alpha) \leq 1$ and $\Sigma_i^N \wp_i=1$. Figure 1 shows a

probabilistic operator that has three subsets of effects (arrow destinations correspond to subsets). Each subset has add-effects and del-effects (propositions) that describe the state modifications generated by the application of the operator with an associated occurrence probability.

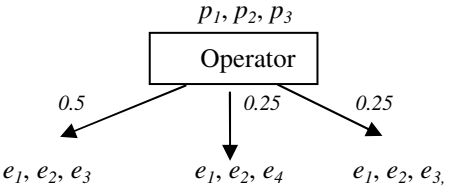


Fig. 1. Generic probabilistic operator

3 Graphplan Framework

The Graphplan planner [17] defines an efficient method of solving classical STRIPS problems. Graphplan is based on compiling a planning problem into a polynomial-size structure called *planning graph*. A planning graph is a layered graph alternating between proposition nodes layers and action nodes layers. Each level represents the union of what might be reachable at a given time step. The initial level consists of proposition nodes that represent the initial conditions. The next level of the graph has a node for each operator that might possibly be performed at the corresponding time step. Special actions are used to represent the persistence of a proposition from one time step to the next. The second proposition level consists of the add-effects of the first action level. Successive levels are generated by inference according to action preconditions and effects. Edges in a planning graph connect actions to their preconditions and their add and delete effects.

During graph building, the graph retains binary mutual exclusion information, indicating when two actions or propositions can not exist simultaneously. This information can be used to prune the search. Graphplan begins by creating a planning graph from the initial conditions until all the goals appear in the graph and none are pairwise mutually exclusive, then searches the planning graph for the plan solutions using a backward chaining search.

3.1 PGraphplan

A structural change to the standard planning graph is required in order to accommodate probabilistic actions. In this subsection we will describe this changes and the search procedure for PGraphplan [8, 9].

We consider probabilistic operators, such as presented in section 2. To support these kinds of operators, the graph is constructed in the normal manner except that each outgoing edge of an action has associated with it the event (subset of effects) which produced the edge.

3.2 Searching the Probabilistic Planning Graph

Given a bounded number of time steps tmax, Pgraphplan builds the planning graph and then performs a finite-horizon dynamic programming to find the optimal *tmax*-step contingent plan [9]. The optimal plan corresponds to the plan of highest expected utility. The utility function adopted in this paper corresponds to finding the plan with the highest probability of success. The complete algorithm and more details about Pgraphplan search can be found in [8, 9].

To speed up the search, Pgraphplan propagates two distinct kinds of information through the graph. Both kinds of information are used to tell the planner when the path it is currently exploring provably cannot reach the goal within the given time horizon and therefore it may safely return failure in its recursive calls [8]. The two types of information are summarized below.

1. Removing unneeded vertices: a "needed" node is the node from which there exists at least one path to the goal. Pgraphplan removes from the planning graph all nodes that do not have any paths to the goal literal (unneeded nodes). This occurs as follow. Are removed from the last propositional level (*tmax* level) of the graph, all propositions that not are in the goal. Now, consider the actions at *tmax*-1 action level. If all add-effects of the one action were removed, then that action too is unneeded and can be withdrawn from the graph. Working backwards, any proposition at time *tmax*-1 that has no out-edges can be removed as well, and so on.
2. Value propagation: the idea of value propagation is to propagate heuristic values (*hvalues*) through the nodes of the graph such that the heuristic value of any state *S*, defined to be the sum of the *hvalues* of the nodes of the states, is guaranteed to be greater than or equal to the true value of *S*. If the planner finds that the heuristic value of the current state at time *t* is less than *t*, then it can backtrack. This means that it cannot possibly reach the goal by time *tmax*.

4 HIPU - Hierarchical Planner Under Uncertainty

HIPU – Hierarchical Planner under Uncertainty [18] is an extension of the Alpine method [15], adapted to act under uncertainty conditions. Regarding uncertainties, it allows a probabilistic distribution of possible operator effects and a probabilistic choice under possible initial states of the domain. HIPU automatically generates an abstraction hierarchy and plan from this hierarchy. Planning begins at the highest level of abstraction, and the solution found in this level is refined by lower levels. During refinement, the plan found so far is evaluated, verifying if the solution succeeds with probability equal or higher than a predefined value.

4.1 Generating Abstractions in HIPU

The starting point for generating abstraction hierarchies in HIPU is Algorithm 1. This algorithm establishes the possible interactions among literals, creating a graph of constraints that will be used in the creation of the abstraction hierarchy.

```
Function Find-Constraints(graph,operators,goals):
Input: The operators of the apace problem and the goals of a problem.
Output: constraints to guarantee ordered monotonicity for the problem.
For each literal in the goals do
If not(Constraint-Determinated(literal, graph))
Constraint-Determinated(literal,graph)←TRUE;
   For each operator in Operators do
         Subset_relevant ←FALSE;
         For each Subset_Effects(operator) do
           If literal in Subset_Effects (operator) do
               Subset_Relevant←TRUE;
               For each Effect in Subset_Effects (operator) do
               Add_Directed_Edge(literal,Effect, graph);
         If (Subset_Relevant)
               preconds←Preconditions(operator);
               For each precondition in Preconds do
                   Add-Directed-Edge(literal, precondition, graph);
               Find_constraint(graph, operators, Preconditions);
return(graph);
```

Algorithm 1. Algorithm for determining constraints

After creating the graph of constraints, we find the strongly connected components using depth-first search. The next step constructs a reduced graph where the nodes that comprise connected components in the original graph correspond to a single node in the reduced graph. Literals within a node in the reduced graph must be placed in the same abstraction space and the connections between nodes define a partial order. The partial order is transformed into a total order using a topological sort. The total order graph originates the abstraction hierarchy, each level of the total order graph corresponding to a hierarchical level.

4.2 Hierarchical Planning Solver

Given a hierarchy of abstractions, HIPU proceeds as follows. First the problem solver maps the original problem to the highest level of the hierarchy (level i) by deleting literals from the initial states, goal states and operators literals that are not relevant to the abstraction level. The planner then performs a depth-first search adapted to probabilistic operators to find a solution for that level. The solution found in level i will be used in the next abstraction level (level i-1), where the literals of the intermediate states serve as sub goals at level i-1. The problem solver then solves each of the intermediate subproblems, using the final state of one subproblem as initial state for the next subproblem. The process is repeated until the plan has been refined in all hierarchical levels.

During the search, when a plan that satisfies the goal state is found, the probability that the current plan will achieve it is computed using the *forward assessment algorithm* [4]. If the probability is high enough, then the plan is a solution and planning terminates successfully, otherwise the planner continues, choosing a new state to expand or returning fault.

5 HPGP – Hierarchical Probabilistic Graphplan

The main difference between HIPU and HPGP is that instead of depth-first search, HPGP uses PGraphplan to find plans. Some modifications are necessary so that PGP can handle the hierarchical planner HIPU.

The first adaptation to embed PGP into the HIPU framework regards the maximum time steps (*tmax*) for the plan search. In PGP the number of time steps is previously defined by the user. This requires knowledge about the domain of the problem or exhausting attempts until that ideal number is found. As HIPU is totally automated, we need to discover a manner to automatically define *tmax*. We propose an alternative using the planning graph. First we expand the probabilistic graph until all the goals appear in it. While the goal state is not reached, the graph expansion continues until it is leveled (two consecutive levels are identical), or a maximum steps number (a system attribute) is reached. When a graph is leveled without reaching all the goals, it means that no solution exists. The next step is to remove from the graph all unneeded vertices and attribute to *tmax* the number of different actions (minus the persistent ones) existing in the graph. Although this heuristic seems reasonable, it can be conservative in specific cases, because the idea is to define *tmax* as the minimum number of operators necessary to reach the goal state. For example, consider the Blocks World problem presented in Appendix A. The operator *paint_block* reaches the goal "painted block" with probability 0.8, but there exists a probability of finishing the paint with the block not painted. However, if we carry the paint and try to paint the block again, the probability of concluding the task with success increases. Certainly, these two new steps (operators) will not be included in the plan if we expand the graph until the level of the goal state level is reached. Another situation in which the heuristics can be conservative is when an operator removes preconditions of other operators, making it necessary to include or repeat actions. But this situation happens because the probabilistic graph plan is a relaxed model (that ignores delete effects), and not exactly a consequence of the heuristics adopted for the *tmax* calculation. This problem could be solved with the inclusion of mutual exclusion relations in the graph, as suggested in [20]. Nevertheless, in this paper we ignore this improvement. When the method to define *tmax* fails during planner execution, the user will be required to inform *tmax*, or it will be given the maximum value allowed.

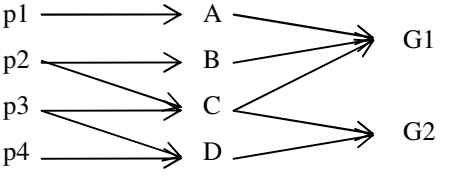


Fig. 2. Operators with redundant effects

Contrasting with previous cases (where the heuristic is conservative), we now discuss a situation in which the heuristic will overestimate *tmax*, causing a search space augmentation. The value *tmax* would be overestimated if there are redundant

propositions in the operator effects. An example is illustrated in Figure 2. The graph showed has four actions (*A, B, C* and *D*), three of these (*A, B* and *C*) reach a goal literal G1 and two (*C* and *D*) goal G2. In the plan, just one of them will be really executed (action *C*). As each action has add-effects linked to a goal literal, none of them will be deleted during removal of the unneeded nodes, causing a increase (from 1 to 4) in the number of time steps (operators) to reach the goal state. This problem would be attenuated by considering only one operator when more than one reaches the same propositions.

A key difference between HIPU and PGP is the type of proposed solution. HIPU solves the problem by finding a sequence of operators (concatenation of partial plans) that reach the goal state with an associated probability of success. PGP creates an optimal contingent plan (an optimal finite-horizon policy), i.e., a mapping from states to operators with the goal of maximizing an expected utility for some utility function (probability of success). Those search models are incompatible. We now propose two alternatives to solve this problem, as follows.

The first alternative modifies HIPU to search contingent plans and not sequential plans. This implies refining all the valid plans found during the search. However, working with contingent plans would cause an increase in the plan space, and we cannot guarantee that it will lead to a significant reduction through hierarchization. We consider then a second solution: to refine the plan with the highest success probability. If the plan can't be completely refined, the second plan with highest probability of success will be chosen, and so forth. Algorithm 2 summarizes HPGP.

Performs HPGP (Initial State, Goal State, Operators):

1. Generate the abstraction hierarchy exactly as HIPU.
2. Create the abstraction spaces for each hierarchical level by deleting from the initial state, goal states and operators propositions that are not relevant to the abstraction level.

For each hierarchical level and respective subproblems:

3. Produce the probabilistic planning graph.
4. Remove unneeded nodes.
5. Calculate the number of actions in the graph and attribute the value to *tmax*.
6. If it is not possible to reach the goal state in the graph, backtrack (return fail), else continue.
7. Reinsert the temporary excluded nodes in the graph.
8. Continue the expansion until the level *tmax* is reached.
9. Exclude unneeded nodes and propagate heuristic values.
10. Perform a finite-horizon dynamic programming to find the optimal contingent plan (see complete algorithm in [8]).
11. Choose the plan with highest probability of success.
12. Refine the plan found (refinement is the same performed by HIPU, presented in Subsection 4.2) executing steps 3 to 12 for each subproblem.

Algorithm 2. High level HPGP algorithm

6 Empirical Results

HPGP was partially implemented. The code was generated in C (for planning) and Lisp (for hierarchy determination). We performed experiments for the Blocks World and Flat-Tire Domains (Appendix A), and compared results with Pgraphplan, HIPU,

and standard top-down Dynamic Programming TDDP [8]. HIPU executes standard top-down Dynamic Programming for searching plans, and considers the same time horizon (*tmax* values) as HPGP.

For the first case study (Blocks World Domain), the abstraction hierarchy generated by HIPU (or HPGP) contains two levels and the respective literals: Level 1 - Paint and color(x), and Level 2 - on(x,y), onTable(x), clear(x) and clear(y). Numerical results are in Table 1. The number of states of HIPU and HPGP is the sum of states generated at each abstraction level.

For the experiments presented in Table 1, *tmax* was automatically calculated using the heuristic discussed in Section 5. Table 2 shows results with a manual definition of *tmax*. Only Pgraphplan and top-down Dynamic Programming (contingent planners) were considered in these experiments. Results are compared in terms of the number of states searched and the probability of success of the generated plans.

Table 1. Results in the Blocks World Problem, 4 and 7 blocks, t*max* automatically generated

Planner	**States**		**Probability**	
	4 blocks *tmax* = 9	7 blocks *tmax*=18	4 blocks	7 blocks
HIPU	224	123469	0.4096	0.209715
HPGP	90	56037	0.4096	0.209715
Pgraphplan	787	146652	0.4096	0.209715
TDDP	12507	>33600000	0.4096	0.209715

Table 2. Results in the Blocks World Problem, 4 and 7 blocks, t*max* user-defined

Planner	**States**		**Probability**	
	4 blocks *tmax*= 12	7 blocks *tmax*=22	4 blocks	7 blocks
Pgraphplan	7547	20229508	0.73728	0.738198
TDDP	23164	Timeout	0.73728	Timeout

HPGP empirically searches through less states than HIPU. Both hierarchical planners have advantages (in search states) when compared with Pgraphplan and TDDP. Contingent planners (Pgraphplan and TDDP) are able to find plans with higher probability of success when we increase the number of time steps (*tmax)*, but with a higher cost (number of states searched).

Table 3. Results in the Flat-Tire domain, t*max=14*, automatically calculated

Planner	**States**	**Probability**
HIPU	2657	0.8145
TDDP	6350	0.8145
HPGP	1213	0.8145
Pgraphplan	2886	0.8145

Experiments realized in the Flat-Tire domain are showed in Table 3. Hierarchization divided the problem in two hierarchical levels, with respective literals: Level 1 - on(nut,hub), on-ground(nut), free(hub), in(wheel,container), have(wheel), on(wheel,hub). Level 2 – open(container), close(container). The number of time steps *tmax* was automatically calculated, it is the same (14) for all planners.

HIPU and TDDP use the same search strategy, however, the hierarchical process (HIPU planner) produces better results than the flat planner (TDDP). HPGP was significantly better (less search states) than HIPU and Pgraphplan.

7 Conclusions and Future Work

This paper presented HPGP, an abstraction-based framework for probabilistic planning. The abstraction hierarchy is automatically generated by a hierarchical planner under uncertainty, and the plan search proposed here is an extension of the Pgraphplan planner, adapted to handle hierarchical planning. We proposed a heuristic to automatically generate time steps (*tmax*) to finite-horizon search.

We related preliminary experimental results on the Block's World and Flat-Tire Domains. An analysis of the results demonstrates that HPGP can reduce significantly the space search in probabilistic planning. However, to guarantee that the heuristic that automatically defines *tmax* is viable would be premature. We prefer to leave the definition of the number of steps as an open question until improvements (some of which are proposed in this article) are implemented.

Proposals for future work are adapting HPGP to search contingent plans and executing Tgraphplan [8] to search plans in our hierarchical framework. Tgraphplan finds the highest probability trajectory from the start state to the goal, and produces potentially sub-optimal policies. We are working at the implementation of a future Hierarchical Tgraphplan Planner (HTGP). Likewise, a deeper evaluation of HPGP performance through comparisons with other planners and tests on other domains are activities for future research.

We also suggest using different search strategies and heuristics to increase the hierarchical planner efficiency.

References

1. Geffner, H., Bonet, B.: Solving Large POMDPs Using Real Time Dynamic Programming. In: Working Notes Fall AAAI Symposium on POMDPS (1998)
2. Bonet, B., Geffner, H.: Planning and Control in Artificial Intelligence: A Unifying Perspective. Appl. Intell. 14(3), 237–252 (2001)
3. Younes, H.L.S., Littman, M.L.: PPDDL1.0: An extension to PDDL for expressing planning domains with probabilistic effects. Technical Report CMU-CS-04-167, Pittsburgh (2004)
4. Kushmerick, N., Hanks, S., Weld, D.: An Algorithm for Probabilistic Planning. Artificial Intelligence. 76, 239–286 (1994)
5. Blyte, J.: Planning Under Uncertainty in Dynamic Domains. Doctor Thesis. Carnegie Mellon University, Pittsburgh, PA (May 1998)

6. Bougerra, A., Karlsson, L.: Hierarchical Task Planning under Uncertainty. In: 3rd Italian Workshop on Planning and Scheduling, Perugia, Italy (2004)
7. Doan, A., Haddawy, P.: Decision-Theoretic Refinement Planning: Principles and Application. Technical Report TR 95-01-01 (February 1995)
8. Blum, A.L., Langford, J.C.: Probabilistic planning in the graphplan framework. proceedings EPC, 319–332 (1999)
9. Blum, A.L., Langford, J.C.: Probabilistic planning in the graphplan framework. In: AIPS 1998 Workshop on Planning as Combinatorial Search (June 8-12, 1998)
10. Little, I., Thiébaux, S.: Concurrent Probabilistic Planning in the Graphplan Framework. In: ICAPS 2006. Proceedings: 16th International Conference on Automated Planning and Scheduling (2006)
11. Little, I., Aberdeen, D., Thiébaux, S.: Prottle: A probabilistic temporal planner. In: AAAI 2005. Proc. of the 20th American Nat. Conf. on Artificial Intelligence (2005)
12. Nau, D., Cao, Y., Lotem, A., Munoz-Avila, H.: Shop: Simple hierarchical ordered planner. In: Procs. of the Int. Joint Conference on AI, pp. 968–973 (1999)
13. Giunchiglia, F., Villaorita, A., Walsh, T.: Theories of abstraction. AI Communication 10(3-4), 167–176 (1997)
14. Knoblock, C.: An analysis of Abstrips. In: Artificial Intelligence Planning Systems: Proceedings of the first international conference (1992)
15. Knoblock, C.: Automatically generating abstractions for planning. Artificial Intelligence 68(2), 243–302 (1994)
16. Armano, G., Cherchi, G., Vargiu, E.: A Parametric Hierarchical Planner for Experimenting Abstraction Techniques. In: IJCAI 2003, pp. 936–941 (2003)
17. Blum, A., Furst, M.: Fast Planning through Planning Graph Analysis. Artificial Intelligence 90, 281–300 (1997)
18. Friske, L.M., Ribeiro, C.H.C.: Planning Under Uncertainty with Abstraction Hierarchies. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 1057–1066. Springer, Heidelberg (2006)
19. Ghallab, M., Nau, D., Traverso, P.: Automated Planning Theory and Pratice. Morgan Kaufmann Publishers, San Francisco (2004)
20. Wenxiang, G., Rixian, L., Huajie, O., Minghao, Y.: An Improved Probabilistic Planning Algorithm Based on Pgraphplan. In: Proceedings of the Third International Conference on Machine Learning and Cybernetics, vol. 4, pp. 2374–2377 (2004)

Appendix A

Blocks World Domain. augmented with probabilistic *colorblock* operator. This domain consists of a set of cubic blocks sitting on a table. The blocks can be stacked, but only one block can fit directly on top of another. A robot arm can pick up a block, paint the block, and move it to another position, either on the table or on top of another block. The arm can only pick up one block at time, so it cannot pick up a block that has another one on it. The goal will always be to build one or more stacks of painted blocks, specified in terms of what blocks are on top of what other block. This domain has four operators, as follow:

Op: stack (*x y*), Preconds: clear (x), clear (y), on-table (x), Effects: on (x y), ~clear(y), ~on-table(x)
Op: unstack (*x y*), Preconds: on(x y), clear (x), Effects: on-table(x), clear(y), ~on(x y)

Op: colorblock(*x*), Preconds: clear(x), on-table(x), paint, no-color(x), Effects: 0.8(color(x), ~no-color(x)), 0.2(~paint, no-paint)
Op: chargepaint, Preconds: no-paint, Effects: paint, ~no-paint.

Flat-Tire Domain considers the problem of fixing a flat tire. The domain has the following eight action schemata:

Op: open(*x*), Preconds: closed(x), Effects: ~closed(x), open(x)
Op: close(*x*), Preconds: open(x), Effects: ~open(x), closed(x)
Op: fetch (*x y*), Preconds: in(x y), open(y), Effects: ~in(x y), have(x)
Op: put-away(*x y*), Preconds: have(x), open(y), Effects: in(x y), ~have(x)
Op: remove-wheel(x y), Preconds: on(x y), on-ground Nut1, on-ground Nut2, on-ground Nut3, on-ground Nut4, Effects: have(x), free(y), ~on(x y)
Op: put-on-wheel(x y), Preconds: have(x), free(y), Effects: on(x y), ~free(y), ~have(x)
Op: remove-nut(x y), Preconds: on(x y), Effects: 0.95(~on(x y), on-ground(x))
Op: fixit-nut(x y z),Preconds: on-ground(x), on(z y), Effects: ~on-ground(x), on(x y)

Correction of Medical Handwriting OCR Based on Semantic Similarity*

Bartosz Broda and Maciej Piasecki

Institute of Applied Informatics, Wrocław University of Technology, Poland
{bartosz.broda,maciej.piasecki}@pwr.wroc.pl

Abstract. In the paper a method of the correction of handwriting Optical Character Recognition (OCR) based on the semantic similarity is presented. Different versions of the extraction of semantic similarity measures from a corpus are analysed, with the best results achieved for the combination of the text window context and Rank Weight Function. An algorithm of the word sequence selection with the high internal similarity is proposed. The method was trained and applied to a corpus of real medical documents written in Polish.

Keywords: semantic similarity, handwriting, OCR correction, Polish.

1 Introduction

Contemporary medical documents are created mostly in electronic form, but thousands of medical documents created and stored in archives are a possible source of very valuable knowledge. The task of off-line recognition of medical handwritten documents is distinguished from the general handwriting optical character recognition (OCR) by two important aspects. The task seems to be easier as documents come from a known source, e.g. a hospital and the domain and the group of authors are limited. On the other hand, medical handwriting is commonly recognised as an example of an almost illegible handwriting.

The difficult task of recognition on the level of letter images can be supported by the prediction of letter and word sequences on the level of language modelling. The problem of letter sequence modelling seems to be well investigated and solved in the case of the particular system discussed in the paper, e.g. [1]. However, on the level of word sequences the methods of morpho-syntactic modelling and stochastic language models did not give a fully satisfying solution. The former had too low accuracy, the latter were too sensitive for the differences between the learning and testing corpora. Moreover, both approaches do not utilise the semantic information for the OCR correction. The aim of this work is the construction of handwriting OCR correction algorithm utilising the information concerning the semantic properties of words. The method should not depend on any manually constructed semantic resource — all information should be directly extracted from a learning corpus of text.

* This work was financed by the Ministry of Education and Science, the project No 3 T11E 005 28, and partially by the Inst. of Applied Informatics WUT.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 437–446, 2007.

2 Semantic Model for Correction of OCR Output

2.1 Task Formulation

The main task of the constructed OCR system is to transform a handwritten medical document into its electronic version. We assume that documents come from one institution (e.g. a hospital ward), that limits their domain, and that they are written in Polish with some possible addition of foreign words. The whole system is divided into into three main parts (layers) [1]:

- a *character level* — soft recognition of isolated characters; for each character, a subject of recognition, the classifier fetches the vector of support factors for all characters in the alphabet,
- a *word level* — works on the results of character classification, isolated words are recognised using also soft classification paradigm: Hidden Markov Models (HMM) and Probabilistic Lexical Language Models (PLLM) combined with a classifier based on an incomplete probabilistic lexicon; the combined character level and word level are called a *word classifier* (WC),
- a *language modelling level* based on the rejection approach applying a *language model* (LM) to lists of *candidates* generated by WC, where a candidate is a possible recognition for a word position in the input text.

Our experiments were performed on the basis of *The Medical Corpus of the Institute of Applied Informatics* (KorMedIIS) [2] of electronic medical texts that was collected from the database of a hospital for which the prototype OCR system is being constructed. The collected texts belong to several categories but only The Corpus of Epicrises (CE) was used in the experiments. An epicrisis is short descriptions of a patient stay in a hospital, consists of several sentences (phrases), reports some details of the patient stay and treatment, and often copies after the other documents. CE includes 15 251 epicrises (1 701 313 words). During experiments, two testing parts of CE were randomly selected: TC1 including 300 epicrises (32 694 words) and TC2 with 2 240 epicrises (206 662).

As we were not able to collect a sufficiently numerous set of hand-written texts, a simulated experiment was performed. Text images were artificially created using the set of images of 5 080 hand-written characters, manually classified. A text image is created in the following steps. First, the text to be recognised is randomly drawn from the Test Corpus. Next, for each character one image of this character is randomly selected from the set of character samples. Finally, the drawn character images are arranged side by side into an artificial text image.

The simplified version of the WC used in experiments assumes that word images are correctly segmented into isolated characters. However we do not deal here with the non trivial problem of word segmentation, but we rather focus our attention on the efficient construction of a LM for the domain of medical texts written in an inflective language, e.g. Polish. For each written word in the text, the WC produces a list of the $k = 10$ most probable *candidate words* (henceforth *candidates*). Each candidate is assigned a *score*, i.e. a number which denotes how likely that word was correctly recognised according to the used classifier. In the

case of punctuations and numbers it is assumed that the WC produces perfect recognitions and returns only one candidate. The task of the LM is to select the proper recognition among the k possible for each position.

In [1], a morpho-syntactic LM was applied to lists of the k-best candidates independently of the WC scores. Next both classifiers were combined. The n-gram LMs of [3] were applied to the k-best lists, but WC scores were not used. The best candidate sequence was identified among the k-best ones. As the Viterbi search used in [3] produces the best path across the candidates, but gives no ranking of candidates on the subsequent positions, there is no simple way to combine the LMs of [3] with the scores of the WC. Contrary to this, we assumed here a gradual improvement of the result of the WC. In that way we want to omit the merging problems encountered in [3] and partially in [1], too. We want to explore WC scores in a style of the *grey-box* approach: not depending on the exact mechanisms of the WC, but taking into account its characteristics.

In order to make more space for the improvement introduced by the LM, we used in all experiments a version of the WC with the decreased accuracy. The WC achieves accuracy of 86.05% of words correctly recognised when the first candidate form the list is taken (96.69% in $k = 10$ candidates) as calculated for all *input tokens* from the TC1 corpus (86.09% and 96.37% respectively for TC2). The accuracy measured for *ambiguous tokens* (WC returned more than one candidate with the score above 0) is respectively: 80.2% and 95.31% ($k = 10$) for TC1 and 80.39% and 94.89% ($k = 10$) for TC2. We distinguished also *tokens without the answer* (TWA) among candidates. After testing the accuracy of the WC for the different values of k, we decided to identify a cut-off threshold τ for scores in order to balance the number of candidates analysed by the LM and the maximal possible accuracy. Two parameters were defined:

- a *cut-off precision* — the precision of recognition calculated only for ambiguous tokens, excluding TWAs, with at least one candidate above τ,
- a *coverage* — the number of candidates below τ in relation to all candidates from ambiguous tokens.

The cut-off precision determines the highest possible accuracy of decisions to be achieved by the LM. The coverage informs about the decrease in the number of candidates to be analysed by LM. The relation of the cut-off precision, coverage and threshold values for TC1 is presented in Fig 1 (for TC2 this relation is similiar). On this basis we selected $\tau = 0.85$, that corresponds to the cut-off precision $\approx 99\%$ and coverage $\approx 43.9\%$, i.e. less than half of the candidates are eliminated from tokens for which decission must be made.

2.2 Models

The idea is to select candidates, one per token, such that they form a group of the maximal average pair-wise *semantic similarity*. The measure of semantic similarity is extracted from the training corpus, see Sec. 2.3. In that way, a sequence of semantically related candidates is chosen as the recognition. Only

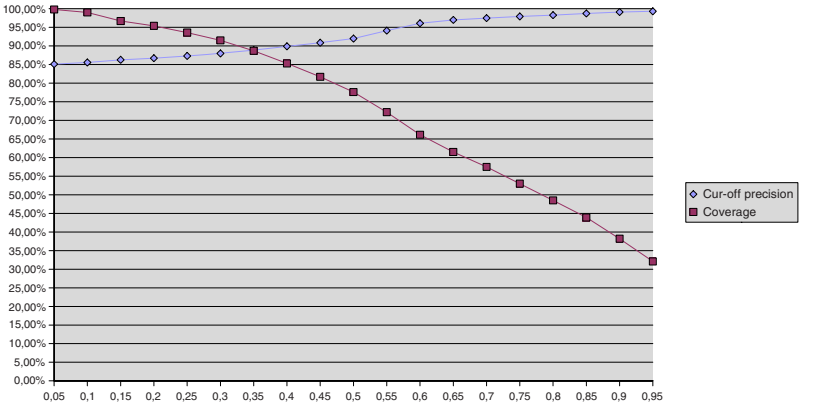


Fig. 1. The relation of the cut-off precision, coverage and threshold values

candidates above the defined threshold τ are analysed. The general scheme of the algorithm is based on the *k–means algorithm* [4]:

– For each *text context* (a *document* or a *text window*):
  1. An initial centre of the cluster — a mean vector, is set.
  2. For each token a candidate which is the most similar to the mean vector is chosen.
  3. The mean vector is recalculated according to the chosen candidates.
  4. `If` a stop condition is not fulfilled `go to` 2.

Each candidate is represented by a vector of real values in the (possibly transformed) coincidence matrix (see Sec. 2.3). The initial mean vector is calculated in the step 1 as the average of the vectors of the first candidates of these tokens, which are unambiguous after the τ elimination. Moreover, because of the numerous *ad hoc* abbreviations and symbols, unambiguous tokens of less than three letters, e.g. "*ż.*" — the *ad hoc* abbreviation of "życia" ($life_{case=gen}$) are not taken into account during the initial centre calculation. If there is no unambiguous token, one token is randomly chosen and its best candidate is taken as the initial mean. If a text window is used as a context, then for each next position of the window, the initial mean vector is a sum of the average with the previous mean vector — we try to keep consistency across the whole document.

For each ambiguous token we select a candidate which is the most *similar* to the mean vector. Different measures of similarity were tested, see Sec. 2.3. Next, the mean vector is modified to the average of all selected candidates. The process is repeated until some number of iterations is reached or the increase of the average similarity in relation to the best level so far is below some threshold.

After the first experiments, see Sec. 3, we noticed that smaller text windows result in better accuracy. Moreover, the very positive results of the application of word n-gram LMs to this task [3] show that the short distance associations among tokens are very important here. Thus, we proposed a new LM based on

semantic similarity called the *Semantic Window* LM (SemWnd). In SemWnd, a *flexible* text window of the minimal size N is moved across a document in left to right direction. The centre of the window is positioned on ambiguous tokens. Next, the borders of the window are gradually extended to the left and right until $N/2$ *content tokens* are collected on both sides. A content token is an input token that can possibly be a good discriminator. Because of the similarity measures used (similarity is calculated for every pair of tokens) here we define content token as a token that has the best candidate of more than 3 letters.

The score of a candidate c_i of the ambiguous token in the centre of the text window is calculated according to SemWnd as following:

$$score(c_i)_{SW} = \sum_{N \geq j \geq 1} sim(c_i, best\_of(t_j)) \tag{1}$$

where t_j is a token in the window, and $best\_of$ returns its best candidate.

The best candidates in the left part of the window, in the case of ambiguous tokens, are defined according to the value of $score()_{SW}$, in the right part according to the scores of the WC. In the case of odd values of N, we take the left part bigger, as in the opposite case the results of the experiments were lower. The accuracy of SemWnd is higher than the accuracy of the WC alone.

2.3 Semantic Similarity for Words

As there is no thesaurus covering the terminology of KorMedIIS, the used *Semantic Similarity Measures* (SSMs) are extracted directly from the corpus. According to the *Distributional Hypothesis* of Harris [5] words occurring in similar contexts have similar meanings. If we are interested only in SSM, then the best results can be achieved when contexts are described by morpho-syntactic features of word occurrences, e.g. [6]. However, in the case of the OCR correction, we need a SSM working for all tokens occurring on the input, including non-words and any symbols. They have no morpho-syntactic properties. Moreover, we want to predict occurrences of particular word forms as following other word forms. Thus, we decided to use tokens (word forms and symbols) as elements being described and to identify the context with co-occurrence of a described token t_i with some other token t_j in a short text window of 5-15 tokens. A *coincidence matrix* of the form $\mathbf{M}[t_i, t_j]$ is created from the learning part of KorMedIIS, where t_i,t_j are two tokens, and the cell contains the number of co-occurrences of t_i and t_j in the same text window with t_i is in its centre. The rows of $\mathbf{M}$ correspond to content tokens (see Sec. 2.2) occurring in the learning corpus, while the columns initially corresponded to all tokens, including non-content ones, too. After the first experiments, we limited the column tokens (i.e. features) only to the ones occurring at least min_{tf} times in the corpus with $min_{tf} = 3$ defined experimentally. Elimination of columns for non-content tokens resulted in the decreased accuracy of the model as non-content tokens express some description of contexts. We did not apply any weight function to the frequencies to preserve the possibility of the probabilistic interpretation of the data.

According to collected earlier experience [7,6], we tested three possible SSMs:

- the *cosine measure* (Cos) applied to row vectors transformed previously by the *logarithmic scaling* and *entropy normalisation* [7],
- *Information Radius* (*IRad*): $D(p||\frac{p+q}{2}) + D(q||\frac{p+q}{2})$, and similarity: $Sim_{IRad} = 10^{-\beta IRad(p||q)}$, where p and q are the probability distributions calculated for the two row vectors being compared, $D(p||q) = \sum p \log \frac{p}{q}$ is Kullback-Leiber divergence, $p(t_i, t_j) = \frac{\mathbf{M}[t_i,t_j]}{TF(t_i)}$, $TF(t_i)$ — the total frequency of t_i in the learning corpus [8], we choose $\beta = 10$ like in [7].
- and the cosine measure applied to vectors produced by the *Rank Weight Function* (RWF) proposed in [6], discussed shortly below.

The aim of RWF is to identify a set of the most descriptive features (matrix columns) for a given row vector $\mathbf{M}[t_i, \bullet]$ (word), and to describe $\mathbf{M}[t_i, \bullet]$ with the ranking of the features in place of the exact feature values [6]:

1. Weighted values of the cells are recalculated using a weight function f_w: $\forall_{t_j} \mathbf{M}[t_i, t_j] = f_w(\mathbf{M}[t_i, t_j])$.
2. Features in a row vector $\mathbf{M}[t_i, \bullet]$ are sorted in the ascending order on the weighted values.
3. The k highest-ranking features are selected; e.g. $k = 1000$ works well.
4. For each selected feature t_j: $\mathbf{M}[t_i, t_j] = k - rank(t_j)$.

Following [6], as the weight function f_w, we applied the *t–score* measure of statistical significance of the pair frequency: t_i and t_j.

2.4 Heuristic rules

Documents of KorMedIIS as written during the real practice in a hospital, are full of short phrases, *ad hoc* abbreviations, and lists of activities or medicines. The 'non-standard' parts are especially difficult for the LMs. Thus we introduced *heuristic rules* correcting some specific cases which were manually identified in the learning data. Two types of rules were defined: *simple rules* and *morpho-syntactic rules*. Simple rules deals with mistakenly recognised short symbols and repair some constant errors made by the WC, e.g. a rule choosing 'x' in the case of the ambiguity between 'x' and 'k' (the WC prefers 'k', what is wrong in the vast majority of cases). Several simple rules were defined.

Morpho-syntactic rules refer to the morpho-syntacic features of candidates which are obtained from the morphological analyser *Morfuesz* [9]. In the case of Polish, an inflective language, the features constrain the possible sequences of candidates. The rules express constraints that should be preserved by proper candidate sequences. Candidates fulfilling the constraints are preferred. Only several rules were constructed so far, as most candidates are ambiguous according to their morpho-syntactic description (an intrinsic property of the inflective language) and the documents contains quite numerous grammatical and spelling errors. The examples of rules are: the rule testing presence a case required by a

preposition or the rule testing the possibility of the morpho-syntactic agreement between a noun and an adjective which precedes it.

The rules are applied during selection of candidates for a token in the centre of the text window. From the candidates fulfilling the constraint the one with the highest $score_{SW}$ (1) is chosen. In the case there is no candidate matching, all are evaluated by the SMM. All the rules, simple and morpho-syntactic, when applied increase the accuracy by 0.5%.

2.5 Merging with the Word Classifier

The results achieved by SemWnd (see Sec. 3) alone, e.g. 92.69% for the TC1, are comparable with the n-gram model (92.8% for the same fold). However, during the manual inspection of the errors we discovered that in the case of the most serious mistakes of SemWnd, e.g. a candidate unrelated to the context selected or a typo introduced (because of typos present in the CE) the WC scores are mostly opposite to the scores of SemWnd. The both classifiers, which are working on different levels and on the basis of different data, seem to be often complementary in their decisions. Thus, we decided to combine the scores and to use a WC score (from $\langle 0, 1\rangle$) as a scaling factor: $score(c_i)_{WC+SW} = score(c_i)_{SW} * score(c_i)_{SW}$. This simple merge increased the accuracy by about 2%–3%.

3 Experiments

In order to directly compare our present approach with the previous works, namely [1,3], we used the same learning and test corpus — TC1. The best result obtained for the TC1 part used in [3] is 92.80%. Two basic parameters were tested: the size of context during the construction of coincidence matrix and the size of the text window used during selection of candidates for ambiguous tokens. For the construction phase the best result was achieved for context of 10 tokens.

Moreover, we tested two methods: the k-means model as the first one and SemWnd. In both models the best results, all presented in Tab. 1, were obtained by applying SMM based on RWF. During the first experiments with SemWnd only simple rules were used, in the next three rules of both types were applied.

The morpho-syntactic LM of [1] (with some heuristic rules) produced 89.02% accuracy on the same corpus. It means that the best results obtained with the help of SemWnd(10,3), i.e. 93.1% (the reduction of the recognition error by 59.7%) outperforms this result. It is also higher that the result of the n-gram LM equal to 93%. However, the difference is not statistically significant. On the larger corpus, i.e. the whole TC, the accuracy of the n-gram model is 92.8%, while the accuracy of the SemWnd(10,3) is 92.69%, but this difference is not statistically significant, too. After merging with WC, SemWnd achieved 94.64%. On the TC2 corpus the accuracy of the n-gram LM is 91.69%. SemWnd achived lower accuracy of 90,88%, but after merging its accuracy increased to 94.28%.

The other SMMs produced significantly lower accuracy in SemWnd. On the TC1 corpus, the best accuracy of the cosine SMM was 91.3%, and of the IRad was 90.9%. This observation is consistent with the results obtained in [7,6].

Table 1. The overall accuracy [%] of the models (TC1,TC2 — folds of the corpus, TC1(2000) — the first 2000 tokens of TC1)

Method	Corpus	Window size				
		Correction				
		3	5	10	15	document
k-means	TC1(2000)	89.45	91.15	90.65	90.80	88.60
SemWnd+rules	TC1(2000)	93.10	92.80	92.05	91.95	—
SemWnd+rules	TC1	92.69	92.51	91.74	91.49	—
SemWnd+rules	TC2	90.88	90.75	89.82	89.36	—
WC+SemWnd+rules	TC1(2000)	95.10	95.00	95.00	95.00	—
WC+SemWnd+rules	TC1	94.64	94.54	94.37	94.26	—
WC+SemWnd+rules	TC2	94.28	94.21	94.02	93.97	—

4 Related Work

It is very hard to find applications of SSMs to the OCR correction in literature. Most works concern applications in Speech Recognition (SR) or correction of spelling errors. In [10] a SSM extracted by the LSA technique [11] (e.g. using the cosine measure) is applied in SR. This approach depends on the similarity threshold set manually for the given domain. Moreover, LSA limits the number tokens processed according to high memory complexity. A SSM based on the retrieval of co-occurrence frequencies of words in a kind of encyclopedia is proposed in [12]. In [13], only a limited subset of "content words" is processed in SR. A pair-wise similarity (PMI measure applied) of words is calculated in a way similar to [14] based on the manually constructed Roget thesaurus. In [14] a SSM based on the counting the number of relation defined between two words in the Roget thesaurus is used in OCR correction. In that way the possibility of correction is limited only to the words present in the thesaurus. The reported overall accuracy is higher than ours, but the method was applied to printed OCR of texts written in a 'standard language'. Lexico-semantic patterns together with a lexicon are applied in detection and correction of errors in a kind of SR system in [15]. However, the method strongly depends on the costly resources: an ontology and a lexicon with semantic classes assigned to lexemes. In [16], the notion of "a low density language" is introduced, i.e. a language with a limited electronic resources. The sub-language of KorMedIIS can be treated as a low density language, too. They argue that for such languages, the methods working on the level of letter models are better suited. In [17] a notion of lexical chains based on semantic similarity is used in the detection and correction of spelling errors. All words which occurr less times than some threshold and are not similar to any other words are identified as suspect. Next, the different spelling variants of suspect words are tested. The used SSM is based on a thesaurus and the whole approach is limited only to nouns. [18] is another work in the area of the spelling correction and the context sensitive spelling error detection. An approach based on LSA is proposed. The method works exclusively on "confusion sets" i.e. the

sets of the most frequent spelling errors. In the similar [19] the learning process is performed on the basis of errors artificially introduced to documents. The errors are generated according to a manually defined set of rules

5 Conclusions

SemWnd outperforms the morpho-syntactic LM proposed in [7]. The documents of KorMedIIS are very specific. They were written on a computer and often one document contains parts copied from the others. The percentage of typos and *ad hoc* created abbreviations is very high (more than 25% of tokens are not proper words). Thus, the problem is positively biased for the applied n-gram model with a simple Laplace smoothing (almost 'memory-based'). However the merge of SemWnd with the WC produced significantly better results. On the other hand, as we tested the gradual decrease of the learning corpus size, we could observe the lower speed of the accuracy decrease in the case SemWnd model, in comparison to the n-gram model. SemWnd expresses the better ability to generalise.

SemWnd model is more flexible in combing it with other classifier in comparison to the n-gram LM. The n-gram LM of [3] produces only the best path across the candidates. It does not return probabilities or scores for candidates. If we want to use that model for the reduction of the candidate lists, we have change the search algorithm to the full search in place of the Viterbi-like search. The time complexity becomes unacceptable. In the case SemWnd there is no such problem — its time complexity is independent on the number of candidates left.

The SSM in SemWnd tends to group different word forms of the same lexeme as semantically similar. Unfortunately, different misspelled variants of a word form are grouped together with it, too. Merging with the WC helps here, but a more general approach should be developed. To increase the accuracy of SemWnd, we need to combine it with a more sophisticated morpho-syntactic analysis of candidate sequences and to develop some spelling correction.

References

1. Godlewski, G., Piasecki, M., Sas, J.: Application of syntactic properties to three-level recognition of Polish hand-written medical texts. In: Bulterman, D., Brailsford, D.F. (eds.) Proc. of the 2005 ACM Symposium on Document Engineering, ACM Press, New York (2006)
2. Piasecki, M., Godlewski, G., Pejcz, J.: Corpus of medical texts and tools. Proceedings of Medical Informatics and Technologies, Silesian University of Technology 2006, 281–286 (2006)
3. Piasecki, M., Godlewski, G.: Language modelling for the needs of OCR of medical texts. In: Maglaveras, N., Chouvarda, I., Koutkias, V., Brause, R. (eds.) ISBMDA 2006. LNCS (LNBI), vol. 4345, pp. 7–8. Springer, Heidelberg (2006)
4. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge (2001)

5. Harris, Z.S.: Mathematical Structures of Language. Interscience Publishers, New York (1968)
6. Piasecki, M., Szpakowicz, S., Broda, B.: Automatic selection of heterogeneous syntactic features in semantic similarity of Polish nouns. In: Matoušek, V., Mautner, P. (eds.) TSD 2007. LNCS (LNAI), vol. 4629, Springer, Heidelberg (2007)
7. Piasecki, M., Broda, B.: Semantic similarity measure of Polish nouns based on linguistic features. In: Abramowicz, W. (ed.) BIS 2007. LNCS, vol. 4439, Springer, Heidelberg (2007)
8. Dagan, I., Lee, L., Pereira, F.: Similarity-based method for word sense disambiguation. In: Proc. of the 35th Annual Meeting of the ACL, Madrid, Spain, ACL, pp. 56–63 (1997)
9. Woliński, M.: Morfeusz — a practical tool for the morphological analysis of Polish [20], 511–520
10. Cox, S., Dasmahapatra, S.: High-level approaches to confidence estimation in speech recognition. Speech and Audio Processing, IEEE Transactions 10(7), 460–471 (2002)
11. Landauer, T., Dumais, S.: A solution to Plato's problem: The latent semantic analysis theory of acquisition. Psychological Review 104(2), 211–240 (1997)
12. Kupiec, J., Kimber, D., Balasubramanian, V.: Speech-based retrieval using semantic co-occurrence filtering. In: Proceedings of the workshop on Human Language Technology, pp. 373–377 (1994)
13. Inkpen, D., Désilets, A.: Semantic Similarity for Detecting Recognition Errors in Automatic Speech Transcripts. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp. 49–56 (2005)
14. Jobbins, A., Raza, G., Evett, L., Sherkat, N.: Postprocessing for OCR: Correcting Errors Using Semantic Relations. In: LEDAR. Language Engineering for Document Analysis and Recognition, AISB 1996 Workshop, Sussex, England (1996)
15. Jeong, M., Kim, B., Lee, G.: Semantic-Oriented Error Correction for Spoken Query Processing. In: 2003 IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 156–161 (2003)
16. Kolak, O., Byrne, W., Resnik, P.: A generative probabilistic OCR model for NLP applications. In: Proc. of the 2003 Conf. of the North American Chapter of the ACL on Human Language Technology, vol. 1, pp. 55–62 (2003)
17. Hirst, G., Budanitsky, A.: Correcting real-word spelling errors by restoring lexical cohesion. Natural Language Engineering 11(01), 87–111 (2005)
18. Jones, M., Martin, J.: Contextual spelling correction using latent semantic analysis. In: Proc. of the 5th Conf. on Applied Natural Language Processing, pp. 166–173 (1997)
19. Al-Mubaid, H., Truemper, K.: Learning to Find Context-Based Spelling Errors. Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques (2006)
20. Kłopotek, M.A., Wierzchoń, S.T., Trojanowski, K. (eds.): Intelligent Information Processing and Web Mining — Proc. of the International IIS: IIPWM 2006. Advances in Soft Computing, Zakopane, Poland, June. Springer, Berlin (2006)

Multiple Classifier Fusion Using *k*-Nearest Localized Templates

Jun-Ki Min and Sung-Bae Cho

Department of Computer Science, Yonsei University
Biometrics Engineering Research Center
134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea
loomlike@sclab.yonsei.ac.kr, sbcho@cs.yonsei.ac.kr

Abstract. This paper presents a method for combining classifiers that uses *k*-nearest localized templates. The localized templates are estimated from a training set using *C*-means clustering algorithm, and matched to the decision profile of a new incoming sample by a similarity measure. The sample is assigned to the class which is most frequently represented among the *k* most similar templates. The appropriate value of *k* is determined according to the characteristics of the given data set. Experimental results on real and artificial data sets show that the proposed method performs better than the conventional fusion methods.

Keywords: Classifier fusion; Decision templates; *C*-means clustering.

1 Introduction

Combining multiple classifiers has been actively exploited for developing highly reliable pattern recognition systems in the past decade [1, 2]. There are two basic parts for generating an ensemble: creating base classifiers and combining the outputs of the classifiers. In order to achieve the higher accuracy of the ensemble, the individual classifiers have to be both diverse and accurate [3, 4]. Two popular methods for creating classifiers are Bagging and Boosting [5]. Bagging creates each individual classifier in the ensemble with a different random sampling of the training set. Thus some instances are represented multiple times while others are left out. In Boosting, examples that were incorrectly predicted by previous classifiers in the ensemble are chosen more often than examples that were correctly predicted.

The outputs of the diverse classifiers have to be combined with some manner to achieve a group consensus. In order to improve further on the performance of the ensemble, several existing and novel combining strategies have been investigated [6, 7]. Some combiners do not require additional training after the classifiers in the ensemble have been trained individually. Majority voting, minimum, maximum, and average are examples of them [8, 9, 10]. Other combiners need training at fusion level. Examples are behavior knowledge space (BKS) [11] and decision templates (DT) [12]. Especially, DT that composes a template for each class by averaging the

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 447–456, 2007.

outputs of classifiers was reported good performance and was used complementarily with a classifier selection method [13]. However, because the DT abstracts the characteristics of a class into a template, there might be the limitation of applying it to complex problems. In our previous work [14], multiple decision templates (MuDTs) which decompose a template into several localized templates using clustering algorithm was investigated to solve this limitation. Since many clustering algorithms rely on a random component, this method would be sensitive to clustering results.

In this paper, we present a novel fusion method, *k*-nearest localized template (*k*-NLT), which refers *k* most similar templates among the multiple decision templates. It may be less affected by clustering results and thus can obtain stable and high accuracy. Finally, to validate the proposed method, its performance are compared with several classifier combining approaches by using real and artificial data sets from the UCI database and ELENA.

2 Background

2.1 Conventional Fusion Methods

Simple fusion methods such as majority voting, minimum, maximum, average, and BKS have been widely used to construct a multiple classifier system.

Majority Voting. For a sample, this method simply counts the votes received from the individual classifiers, and selects the class with the largest number of votes. Ties are broken randomly.

Minimum, Maximum, and Average. These three fusion methods are considered together because they have a similar decision scheme. The minimum method selects the smallest value among the outputs of the classifiers for each class. The minimums are then compared and a class with the larger value is selected. For an *M*-class problem with *L* classifiers, it is calculated as follows:

$$\max_{z=1,\ldots,M} \left\{ \min_{y=1,\ldots,L} \left\{ d_{y,z}(x) \right\} \right\}. \tag{1}$$

Here, $d_{y,z}(x_i)$ is the degree of support given by the *y*th classifier for the sample *x* of the class *z*. The maximum and the average methods are the same as the minimum method except that the biggest values are compared as

$$\max_{z=1,\ldots,M} \left\{ \max_{y=1,\ldots,L} \left\{ d_{y,z}(x) \right\} \right\} \tag{2}$$

for the maximum method, and the average method compares the mean values as

$$\max_{z=1,\ldots,M} \left\{ \operatorname*{avg}_{y} \left\{ d_{y,z}(x) \right\} \right\}, \ \operatorname*{avg}_{y} \left\{ d_{y,z}(x) \right\} = \frac{1}{L} \sum_{y=1}^{L} d_{y,z}(x) . \tag{3}$$

Behavior Knowledge Space. In this method, possible combinations of the outputs of the classifiers are stored in the BKS-table $T \in \{-1, 1\}^{M^L \times L}$. Each entry in the T contains a class label (most frequently encountered amongst the samples of the training data in this cell) or no label (no sample of the training data has the respective combination of class labels). In tests, a new sample can be classified into the label of the entry with the same outputs of the classifiers. It fails to classify when an output pattern is not found in T.

2.2 *C*-Means Algorithm

The *C*-means (or *K*-means) algorithm is an iterative clustering method that finds C compact partitions in the data using a distance-based technique [15]. The cluster centers are initialized to C randomly chosen points from the data, which is then partitioned based on the minimum squared distance criterion

$$I = \sum_{i=1}^{n} \sum_{c=1}^{C} u_{c,i} \left\| x_i - z_c \right\|^2 . \tag{4}$$

Here, n is the total number of samples in the data set, z_c is the center of the cth cluster, and $u_{c,i}$ is the membership of the ith sample x_i in cluster c. The cluster centers are subsequently updated by calculating the average of the samples in each cluster and this process is repeated until cluster centers no longer change. Although this algorithm tends to find the local minima, it is widely used for clustering because of its simplicity and fast convergence.

2.3 Decision Templates

DT proposed by Kuncheva [12] estimates M templates (one per class) with the same training set that is used for the set of classifiers. For the M-class problem, the classifier outputs can be organized in a decision profile as a matrix

$$DP(x_i) = \begin{bmatrix} d_{1,1}(x_i) & \cdots & d_{1,M}(x_i) \\ \vdots & d_{y,z}(x_i) & \vdots \\ d_{L,1}(x_i) & \cdots & d_{L,M}(x_i) \end{bmatrix}, \tag{5}$$

where L is the number of classifiers in an ensemble and $d_{y,z}(x_i)$ is the degree of support given by the yth classifier for the sample x_i of the class z. When decision profiles are generated, the template of the class m is estimated as follows:

$$DT_m = \begin{bmatrix} dt_m(1,1) & \cdots & dt_m(1,M) \\ \vdots & dt_m(y,z) & \vdots \\ dt_m(L,1) & \cdots & dt_m(L,M) \end{bmatrix}, \quad dt_m(y,z) = \sum_{i=1}^{n} u_{m,i} d_{y,z}(x_i) \Big/ \sum_{i=1}^{n} u_{m,i} \tag{6}$$

In the test stage, the similarity between the decision profile of a test sample and each template is calculated. The sample is then categorized into the class of the most

similar template. Kuncheva [16] examined DT with various distance measures, and achieved higher classification accuracies than conventional fusion methods.

3 *k*-Nearest Localized Templates

The DT scheme abstracts features of each class as a template which may be difficult to classify dynamic patterns. For dealing with the intra-class variability and the inter-class similarity of the dynamic patterns, we adopt a multiple template-based approach where patterns in the same class are characterized by a set of localized classification models. Fig. 1 illustrates an overview of the proposed method.

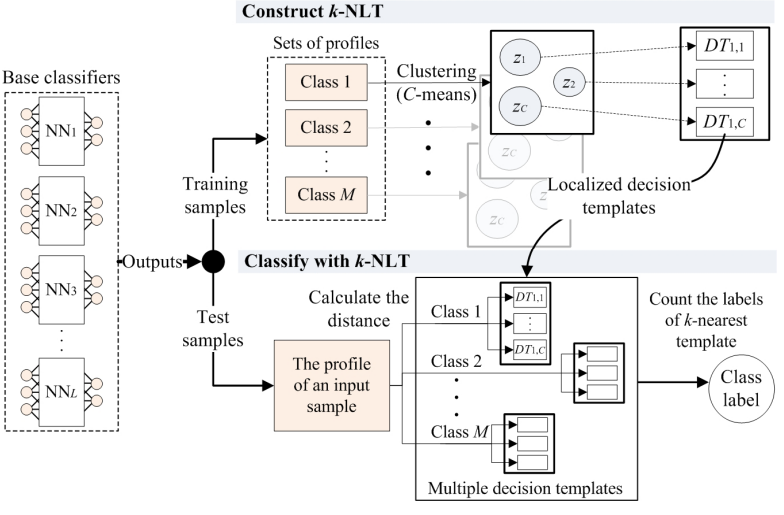


Fig. 1. An overview of the *k*-nearest localized templates

3.1 Estimation of Localized Decision Templates

Localized decision templates are estimated in order to organize the multiple classification models. At first, decision profiles are constructed from the outputs of the base classifiers as Eq. (5) and are clustered for each class using *C*-means algorithm. The localized template of the *c*th cluster in the class *m*, $DT_{m,c}$, is then estimated as follows:

$$DT_{m,c} = \begin{bmatrix} dt_{m,c}(1,1) & \cdots & dt_{m,c}(1,M) \\ \vdots & dt_{m,c}(y,z) & \vdots \\ dt_{m,c}(L,1) & \cdots & dt_{m,c}(L,M) \end{bmatrix}, \quad dt_{m,c}(y,z) = \frac{\sum_{i=1}^{n} u_{m,c,i} d_{y,z}(x_i)}{\sum_{i=1}^{n} u_{m,c,i}} \tag{7}$$

Here, $u_{m,c,i}$ is the membership of the *i*th sample x_i in the cluster c of the *m*th class. Finally, $M \times C$ templates are constructed where M is the number of classes and C is the

number of clusters per class. In this paper the number of clusters was selected as 20 based on the experiments in section 4.1

3.2 Classification Using *k*-Nearest Localized Templates

In the test stage, the profile of a new input sample is matched to the localized templates by a similarity measure. A distance between the profile of a given sample x and the template of each cluster is calculated as follows:

$$dst_{m,c}(x) = \left\| DT_{m,c} - DP(x) \right\|. \tag{8}$$

Since the *C*-means clustering algorithm which was used for generating localized templates is often affected by its random initial instances, it is easy to make error clusters. The error clusters cause a misclassification when the sample is only matched to the nearest template. In order to resolve this problem, the proposed method adopts a *k*-nearest neighbor scheme where the sample is assigned to the class that is most frequently represented among the k most similar templates. In this approach, the appropriate value of k commonly depends on the properties of a given data set. The proposed method, therefore, analyzes the intra-class compactness *IC* and the inter-class separation *IS* (which were originally designed for the validity index of clustering algorithm [17]) of the data set using:

$$IC = E_1 / E_M \ , \ E_M = \sum_{i=1}^{n} \sum_{m=1}^{M} u_{m,i} \left\| x_i - z_m \right\| \tag{9}$$

$$IS = \max_{i,j=1,\ldots,c} \left\| z_i - z_j \right\| \tag{10}$$

where n is the total number of points in the data set, z_m is the center of the mth class, and $u_{m,i}$ is the membership of the ith sample x_i in class m. In this paper we generate a simple rule for k as Eq. (11) based on experiments (see section 4.1).

$$k = \begin{cases} 1 & \text{if} \quad IC \le t_{IC} \ \text{ and } \ IS \le t_{IS} \\ C/2 & \text{if} \quad IC > t_{IC} \ \text{ and } \ IS > t_{IS} \end{cases} \tag{11}$$

4 Experiments

In this paper, we have verified the proposed method on 10 real (R) and artificial (A) data sets from the UCI database and ELENA which are summarized in Table 1. Each feature of data sets was normalized to a real value between -1.0 and 1.0. For each data set 10-fold cross validation was performed. The neural network (NN) was used as a base classifier of an ensemble. We trained the NN using standard backpropagation learning. Parameter settings for the NN included a learning rate of 0.15, a momentum term of 0.9, and weights were initialized randomly between -0.5

and 0.5. The number of hidden nodes and epochs were chosen based on the criteria given by Opitz [5] as follows: at least one hidden node per output, at least one hidden node for every ten inputs, and five hidden nodes being a minimum; 60 to 80 epochs for small problems involving fewer than 250 samples, 40 epochs for the mid-sized problems containing between 250 to 500 samples, and 20 to 40 epochs for larger problems (see Table 1).

Table 1. Summary of the data sets used in this paper

Type	Data set	Case	Feature	Class	Availability	Neural network	
						Hidden	Epoch
R	Breast-cancer	683	9	2	UCI[1]	5	20
R	Ionosphere	351	34	2	UCI	10	40
R	Iris	150	4	3	UCI	5	80
R	Satellite	6435	36	6	UCI	15	30
R	Segmentation	2310	19	7	UCI	15	20
R	Sonar	208	60	2	UCI	10	60
R	Phoneme	5404	5	2	ELENA[2]	5	30
R	Texture	5500	40	11	ELENA	20	40
A	Clouds	5000	2	2	ELENA	5	20
A	Concentric	2500	2	2	ELENA	5	20

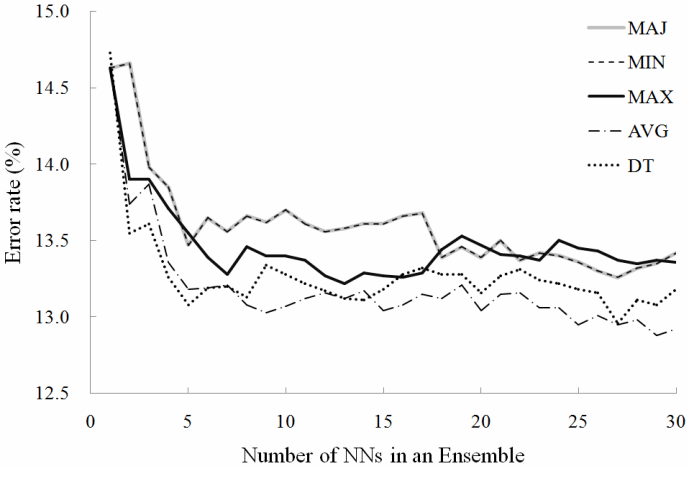


Fig. 2. Average test error over all data sets for ensembles incorporating from one to 30 neural networks

In order to select the appropriate size of an ensemble, preliminary experiments with conventional fusion methods: majority voting (MAJ), minimum (MIN), maximum (MAX), average (AVG), and DT were performed using up to 30 NNs. As shown in

[1] http://mlearn.ics.uci.edu/MLRepository.html
[2] http://www.dice.ucl.ac.be/mlg/?page=Elena

Fig. 2, there is no significant error reduction over 25 classifiers. Therefore, ensemble size of 25 was chosen for the remaining experiments.

4.1 Parameter Setting of the *k*-Nearest Localized Templates

Two major parameters of the proposed method, *C* (the number of clusters per class) and *k* (the number of referring templates), were selected based on the characteristics of given data. The data sets used in our studies were partitioned into two groups according to *IC* and *IS* as depicted in Fig. 3. One group had small values of *IC* and *IS* (Ionosphere, Sonar, Phoneme, Clouds, and Concentric), while the other group had large values of *IC* and *IS* (Satellite, Texture, Segmentation, Breast-cancer, and Iris). In this paper, we chose Ionosphere and Satellite as the representative data sets of the two groups, and performed two series of experiments on them to select *C* and generate the rules for *k* (Eq. 11).

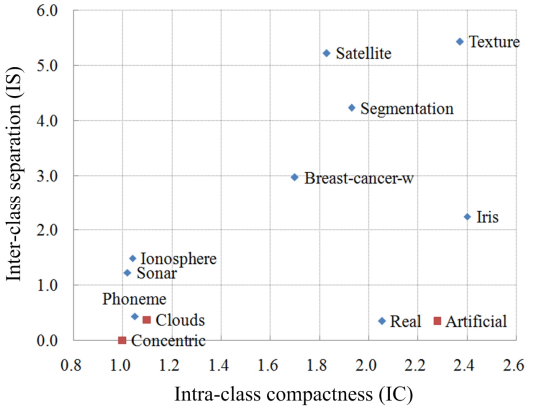


Fig. 3. Characteristics of the data sets used in this paper. *IC* and *IS* are estimated as Eq. (9) and Eq. (10), respectively.

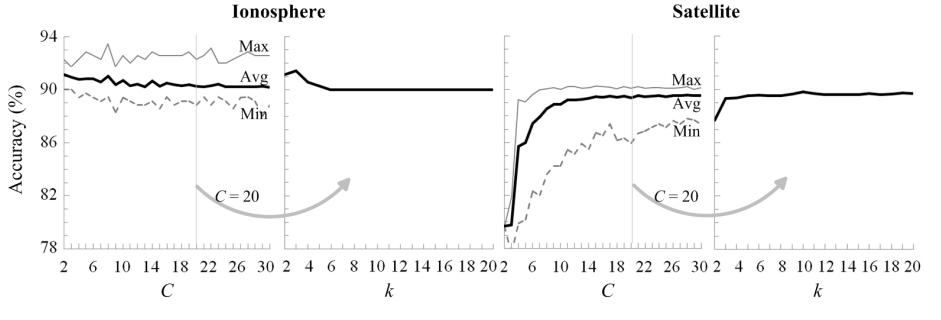


Fig. 4. Accuracies for the two data sets according to *C* (where $k = 1 \sim C$) and *k* (where $C = 20$)

First, we investigated the value of *C* where it had changed from one to 30 while *k* had changed from one to *C*. Since the accuracies were converged after 20 values of *C*,

we fixed C as 20 and changed k from one to 20 in the second series of experiments. As shown in Fig. 4, accuracy was decreased when k was increasing for the Ionosphere. In case of Satellite, on the other hand, accuracy was increased when k was increasing. Therefore, for the remaining experiments, we simply selected k based on Eq. (11) where $t_{IC} = 1.5$, $t_{IS} = 2.0$, and $C = 20$.

4.2 Classification Results

We performed the comparison experiments with k-NLT against the conventional fusion methods. Table 2 provides the accuracies of 10-fold cross validation experiments for all data sets except Ionosphere and Satellite used for the parameter selection of the k-NLT. SB indicates the single best classifier among 25 NNs used in the ensemble. MuDTs, which combine the outputs of the classifiers using localized templates like k-NLT, only refer the class label of the nearest template. Oracle (ORA) was used as a comparative method which is assign the correct class label to an input sample if at least one individual classifier produces the correct class label of the sample. As shown in Table 2, the localized template-based methods (MuDTs and k-NLT) achieved a high classification performance for the overall data sets. Especially, k-NLT showed the best accuracies on more than half of the data sets.

Table 2. Average test accuracy (%) for each data set. Marked in boldface are the best accuracies in each column

Dataset	Breast-cancer	Iris	Segmentation	Sonar	Phoneme	Texture	Clouds	Concentric
SB	97.5 ±1.8	97.3 ±4.7	94.2 ±1.9	85.5 ±6.4	80.4 ±2.0	99.6 ±0.2	79.9 ±3.6	96.2 ±2.7
MAJ	96.9 ±1.6	96.7 ±4.7	94.1 ±1.9	85.5 ±6.4	80.2 ±1.5	**99.7** ±0.2	79.5 ±2.5	97.7 ±1.2
MIN	97.1 ±1.6	96.7 ±4.7	93.6 ±2.2	81.0 ±8.4	80.3 ±1.6	99.6 ±0.2	79.3 ±2.5	97.6 ±1.2
MAX	97.1 ±1.7	96.0 ±4.7	94.4 ±1.9	82.5 ±9.2	80.3 ±1.6	99.6 ±0.3	79.3 ±2.5	97.6 ±1.2
AVG	97.1 ±1.8	**97.3** ±4.7	94.5 ±1.7	**86.0** ±6.6	80.3 ±1.4	**99.7** ±0.2	79.4 ±2.5	97.8 ±0.8
BKS	95.9 ±2.1	93.3 ±8.3	87.7 ±2.8	72.5 ±14.	79.8 ±1.6	97.8 ±0.7	78.6 ±2.4	92.6 ±2.5
DT	**97.2** ±1.8	**97.3** ±4.7	94.5 ±1.7	85.5 ±6.4	80.4 ±1.5	**99.7** ±0.2	79.6 ±2.5	98.0 ±0.8
MuDTs	95.4 ±2.1	95.3 ±5.5	**96.2** ±1.4	84.0 ±7.8	**80.7** ±1.8	99.6 ±0.2	**81.9** ±1.7	**98.8** ±0.6
k-NLT	**97.2** ±1.8	96.7 ±4.7	94.6 ±1.5	84.0 ±7.8	**80.7** ±1.8	**99.7** ±0.2	**81.9** ±1.7	**98.8** ±0.7
ORA	98.7 ±1.8	98.7 ±2.8	98.8 ±0.5	98.0 ±3.5	93.1 ±1.2	99.9 ±0.1	84.7 ±3.7	100 ±0.0

Fig. 5 shows the average test errors and averaged standard deviations over all data sets. The standard deviation can be interpreted as the stability measure of algorithm. BKS showed the worst performance while k-NLT yielded the highest accuracy with stable performance among the compared methods. The paired t-tests between k-NLT and comparable methods, AVG, DT, and MuDTs which produced relatively high accuracies, were conducted and revealed that the differences were statistically significant ($p<0.02$, $p<0.002$, and $p<0.007$ respectively).

In order to compare the relative performance of each method with respect to the others, we calculate a rank score. For each data set, each method was assigned a rank with respect to its place among the others. The highest possible score which assigned to the best model was nine, and the lowest was one. The ranks for each method were then summed to give a measure of the overall dominance among the methods. As shown in Fig. 6, k-NLT achieved highest score among the others.

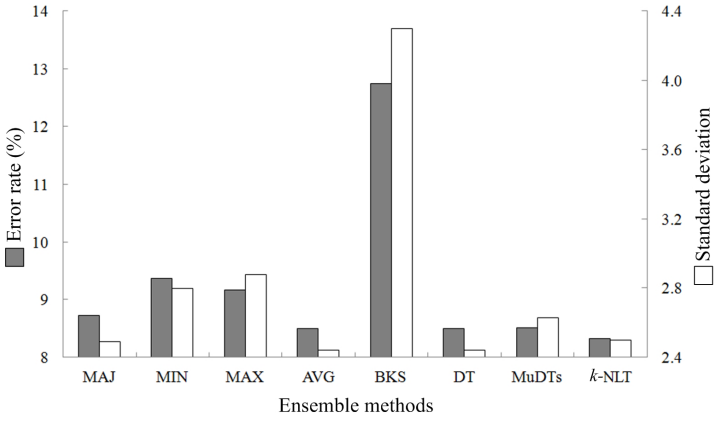


Fig. 5. Average test errors and standard deviations over all data sets used in this paper

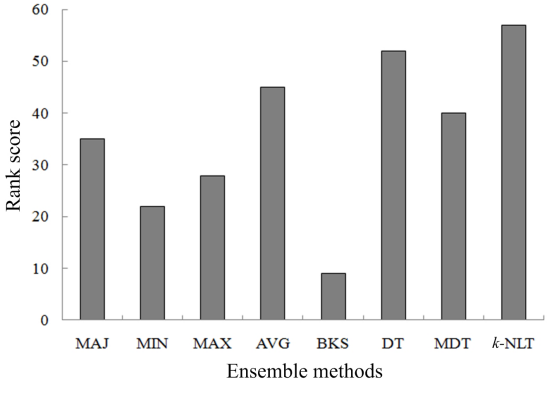


Fig. 6. The sum of rank scores for all data sets. The higher the score, the better the fusion method.

5 Conclusions

In this paper, we proposed *k*-nearest localized templates (*k*-NLT) for combining multiple classifiers. First, decision profiles (the outputs of classifiers) of training set were clustered for each class. Second, localized templates were estimated by averaging decision profiles of each cluster. The templates were then matched to the decision profile of a test sample by a similarity measure. Finally, the sample was assigned to the class which was most frequently represented among the *k* most similar templates. Here, the appropriate value of *k* was selected according to the intra-class compactness and the inter-class separation of a given data set. Experimental results on ten real and artificial data sets showed that the proposed method performed better than conventional fusion methods. The advantage of *k*-NLT can be proved with theoretical backgrounds about the subclass-based model and the *k*-nearest neighbor approach. In

near future, further experiments on additional data sets shall be conducted to analyze the parameters of *k*-NLT.

Acknowledgments. This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the Biometrics Engineering Research Center (BERC) at Yonsei University.

References

1. Cho, S.-B., Kim, J.-H.: Multiple Network Fusion using Fuzzy Logic. IEEE Trans. Neural Networks 6(2), 497–501 (1995)
2. Cho, S.-B.: Pattern Recognition with Neural Networks Combined by Genetic Algorithm. Fuzzy Sets and Systems. 103(2), 339–347 (1999)
3. Hansen, L., Salamon, P.: Neural Network Ensembles. IEEE Trans. Pattern Analysis and Machine Intelligence. 12, 993–1001 (1990)
4. Wanas, N.M., Dara, R.A., Kamel, M.S.: Adaptive Fusion and Co-operative Training for Classifier Ensembles. Pattern Recognition 39(9), 1781–1794 (2006)
5. Opitz, D., Maclin, R.: Popular Ensemble Methods: An Empirical Study. J. Artificial Intelligence Research. 11, 169–198 (1999)
6. Alkoot, F.M., Kittler, J.: Experimental Evaluation of Expert Fusion Strategies. Pattern Recognition Letters 20, 1361–1369 (1999)
7. Kuncheva, L.I.: A Theoretical Study on Six Classifier Fusion Strategies. IEEE Trans. Pattern Analysis and Machine Intelligence 24(2), 281–286 (2002)
8. Lam, L., Suen, C.Y.: Application of Majority Voting to Pattern Recognition: An Analysis of Its Behavior and Performance. IEEE Trans. Systems, Man, and Cybernetics 27(5), 553–568 (1997)
9. Kittler, J.: Combining Classifiers: A Theoretical Framework. Pattern Analysis and Applications 1(1), 18–27 (1998)
10. Kittler, J., Alkoot, F.M.: Sum versus Vote Fusion in Multiple Classifier Systems. IEEE Trans. Pattern Analysis and Machine Intelligence 25(1), 110–115 (2003)
11. Huang, Y.S., Suen, C.Y.: A Method of Combining Multiple Experts for the Recognition of Unconstrained Handwritten Numerals. IEEE Trans. Pattern Analysis and Machine Intelligence 17(1), 90–94 (1995)
12. Kuncheva, L.I., Bezdek, J.C., Duin, R.P.W.: Decision Templates for Multiple Classifier Fusion: An Experimental Comparison. Pattern Recognition 34(2), 299–314 (2001)
13. Kuncheva, L.I.: Switching between Selection and Fusion in Combining Classifiers: An Experiment. IEEE Trans. Systems, Man, and Cybernetics 32(2), 146–156 (2002)
14. Min, J.-K., Hong, J.-H., Cho, S.-B.: Effective Fingerprint Classification by Localized Models of Support Vector Machines. In: Zhang, D., Jain, A.K. (eds.) ICB 2006. LNCS, vol. 3832, pp. 287–293. Springer, Heidelberg (2005)
15. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
16. Kuncheva, L.I.: Using Measures of Similarity and Inclusion for Multiple Classifier Fusion by Decision Templates. Fuzzy Sets and Systems 122(3), 401–407 (2001)
17. Maulik, U., Bandyopadhyay, S.: Performance Evaluation of Some Clustering Algorithms and Validity Indices. IEEE Trans. Pattern Analysis and Machine Intelligence 24(12), 1650–1654 (2002)

Color Image Segmentation Applied to Medical Domain

Liana Stanescu, Dan Dumitru Burdescu, and Cosmin Stoica

University of Craiova, Faculty of Automation, Computers and Electronics, Craiova, Romania
{Stanescu, Burdescu_Dumitru, Stoica_Cosmin}@software.ucv.ro

Abstract. The article presents two practical ways of using the automated color image segmentation in the medical field: for content-based region query and for tracking the time evolution of the disease in patients following a certain treatment. A known technique was used for automated color medical image segmentation – the color set back-projection algorithm. Our previous work in extraction of color regions from a database of nature images using the same algorithm showed promising results. The images are transformed from RGB to HSV color space, quantized at 166 colors and processed by the color set back-projection algorithm that allows the color region detection. The algorithm is studied from two points of view: complexity and the retrieval quality. The experiments that were made on a database with color endoscopy images from digestive tract have shown satisfying results for both applications that are important in practical medical use and medical teaching.

Keywords: image processing, content-based region query, color set back-projection algorithm, HSV color space, color regions.

1 Introduction

Because digital images are produced and used in many domains, the process of retrieving visual information has become an area of intense research during the last years. The initial text-based methods used for finding similar images have been augmented by query-by-example methods that need only to establish a query image without any other helping information. This methodology takes into consideration both coarse granularity and fine granularity preprocessing of the image data. If in the first case, the image is considered as a whole, in the second case the image is processed and the relevant regions are extracted and used for query [10]. In this paper, a fine granularity approach to image indexing and retrieval is adopted.

Medicine is one of the domains where a large quantity of gray-scale or color images, produced by medical devices for establishing a more correct diagnosis, is accumulated. In the medical departments, thousands of images are collected including color endoscopy images, such as the ones used in this paper. In these digestive tract images, the abnormal aspect of the color and texture of the tissue is relevant in indicating a great number of diseases. A series of studies have been made and were created applications that permit medical image processing and the complex query taking into consideration many criteria and characteristics [2], [3], [4], [5] and also automatic image segmentation [9], [10], [11].

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 457–466, 2007.

The applications presented in this paper use a database that is populated by color images captured by an endoscope from a number of patients with stomach diseases. The color set back-projection algorithm is applied on each image for detecting the color regions. The region(s) that represents the sick zone are marked as relevant. The color, the minimum bounding rectangle and the number of pixels that roughly indicates the dimension of the sick region, are stored in the database.

The automated medical color image segmentation has two important uses:

1. in content-based region query
2. during the time evolution of the disease in patients following a certain treatment.

In content-based region query on medical images collections, the specialist chooses one or more of the detected regions for querying the database, with the purpose to retrieve images which are similar by color, texture or both; this can be useful for clarifying some uncertain diagnosis, for seeing the evolution and the treatment in patients with the same diagnosis or in medical teaching.

Concerning the second original use, by applying a certain drugs treatment to some patients, at certain time intervals that are strictly specified by the physician, the images are again collected by the endoscope from the same patients while the same algorithm is performed for detecting the color regions. The relevant region(s) that represents the sick tissue is marked. The comparison between the new and old regions detected as relevant for the same patient, taking into consideration the number of pixels, can help the physician to establish in what percentage the sick region is reduced because of the administrated drugs. This approach may lead to a more rapid and correct estimation of the percentage in which the medication has a good effect in the ulcer diagnosis. Based on the information stored in the database for each patient of the studied lot of sick persons, the statistics about treatment's efficiency can be generated. This may be of help for the patients, specialists, and drugs producers that may intend to test a new medical product.

Motivating factors for our work include:

- There are relatively few published studies for medical color images of the digestive tract.
- Our previous work in extraction of color regions from a database of nature images showed promising results.
- We expect the method to provide a useful, valuable capability for clarifying an uncertain diagnosis or observing the evolution of disease, or observing the regression of disease as a result of treatment; another use is medical teaching, to find additional examples of a particular visual phenomenon.

The paper is structured as follows: section 2 presents the algorithm used for detecting color regions from medical images. Section 3 presents the experiments made and the results obtained during the content-based region query process, and section 4 shows the results obtained in the process of time observation of the disease in patients that follow a certain treatment. Section 5 contains the conclusions.

2 The Color Set Back-Projection Algorithm

For detecting the color regions, it was chosen the color set back-projection algorithm, introduced initially by Swain and Ballard and then developed in the research projects at Columbia University [6]. This technique provides the automated extraction of regions and the representation of their color content. The extraction system for color regions has four steps [6]:

1. the image transformation, quantization and filtering (the transformation from the RGB color space to HSV color space and the quantization of the HSV color space at 166 colors)
2. back-projection of binary color sets
3. the labeling of regions
4. the extraction of the region features

The algorithm reduces the insignificant color information and makes evident the significant color regions, followed by the generation, automatically, of the regions for a single color, for two colors, for three colors.

To conclude with, the second step of the color set back-projection algorithm is the following [6]:

1. Detection of single color regions

1.1 Making the image histogram, H[m], all the values m'=m for which $H[m] \geq p_0$ are detected.

1.2 For each m' the color set c having the property c[k]=1 for k=m and c[k]=0 in other cases is found. On the image R[m,n] the back-projection algorithm for each color set c is applied and the color regions are found. For each region n the local histogram $L_n[m]$ is stored.

1.3 The residue histogram $H_r[m]=H[m]- \sum_n L_n[m]$ is computed.

2. Detection of two colors regions, detection of three colors regions...

For each detected regions the color set that generated it, the area and the localization are stored. The region localization is given by the minimal bounding rectangle (MBR). The number of color pixels represents the region area.

For the implementation of the color set back-projection algorithm, the initial image is transformed in HSV format and quantized. At the end of this process, the color set of the image is obtained [6]. On the matrix that memorizes only the quantized colors from 0 to 165 it is applied a 5x5 median filter, which has the role of eliminating the isolated points. Having the HSV quantized matrix it is possible to begin the process of regions extraction presented above. It may be observed that this process is in fact a depth – first traversal, described in pseudo-code in the following way [1], [12]:

```
FindRegions (Image I, colorset C) is:
InitStack(S)
Visited = ∅
for *each node P in the I do
     if *color of P is in C then
         PUSH(P)
         Visited ← Visited ∪ {P}
```

```
        while not Empty(S) do
              CrtPoint ← POP()
              Visited ← Visited ∪ {CrtPoint}
              For *each unvisited neighbor S of
              CrtPoint do
                   if *color of S is in C then
                          Visited ← Visited ∪ {S}
                          PUSH(S)
                   end if
              end for
         end while
         * Output detected region
    end if
end for
```

The total running time of a call of the procedure FindRegions (Image I, colorset C) is $O(m^2*n^2)$, where "m" is the width and "n" is the height of image.

Proof:

Recall that the number of pixels of the image is m*n, where "m" is the width and "n" is the height of the image. As it is observed next the first loop FOR of the algorithm is executed once at most for each pixel P in the image. Hence, the total time spent in this loop is O(n*m). The WHILE loop processes the stack S for each pixel that has the same color of its neighbor. The inner loop FOR processes the pixels of unvisited neighbor. So, the total time spent in these loops is O(m*n), because all pixels of the image are processed once at most. From previous statements it is inferred that the total running time of this procedure is $O(m^2*n^2)$.

3 Content-Based Region Query - Experiments and Results

In content-based region query, the user selects one or several query regions, trying to find in the database the images containing similar regions from the point of view of color, size or spatial extent.

Taking into account that the color information of each region is stored as a color binary set, the color similitude between two regions may be computed with the quadratic distance between color sets [6].

Other two important distances are taken into consideration: the distance in area between two regions and the distance in MBR width (w) and height (h) between two regions q and t [6]. The single region distance is given by the weighted sum of the color feature $d^f_{q,t}$, area $d^a_{q,t}$ and spatial extent $d^s_{q,t}$ distances. The user may also assign a relative weight α to each attribute. For example, the user may weigh the size parameter more heavily than feature value and location in the query. The overall single region query distance between regions q and t is given by [6]:

$$D_{tot} = \alpha_a \cdot d^a_{q,t} + \alpha_s \cdot d^s_{q,t} + \alpha_f \cdot d^f_{q,t}. \quad (1)$$

Figure 1 shows an image representing gastric cancer diagnosis. The sick part in the image is the prominent oval zone presenting bleedings. The color regions detected by the color set back-projection algorithm are presented in figure 2. From the medical point of view the region 5 and 6 are important.

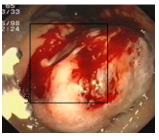


Fig. 1. An image representing gastric cancer diagnosis

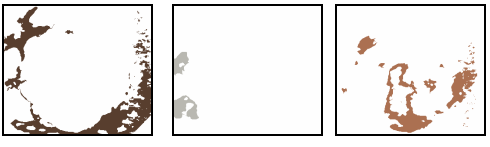


Fig. 2. Color regions detected by the algorithm

The results of the query based on two regions, the Region 5 and 6, are presented in figure 3.

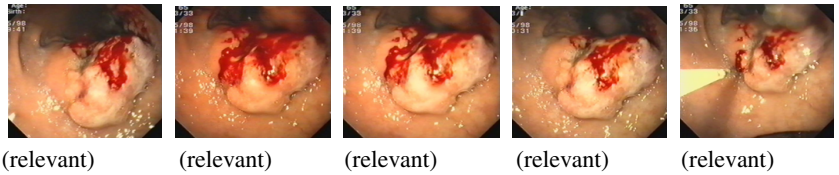


Fig. 3. The results of the content-based query based on the regions Region 5 and Region 6

The query finds 5 relevant images in the first 5 returned images.

The quality of the content-based region query process was tested in the following conditions:

- in the database there are 960 images representing: polyps, colitis, ulcer, ulcerous tumor and esophagitis
- for each image the color regions are detected using the color set back-projection algorithm
- the information about regions (color, area, MBR, position) is stored in the database

- content-based region queries were made on one or many color regions significant from the medical point of view. For each query, the images from the databases were inspected and relevance was assigned to them (1 – relevant, 0 – irrelevant).

In table 1, there are presented some of the results of the queries made. For each experiment the following information is given: the number of query regions, the weights for the distances (color, area and MBR) and the number of relevant images found in the first five retrieved images.

Table 1. The experimental results

Nr. of Query Reg.	Color	Area	MBR	Nr. of relevant images
1	100	0	0	4
1	100	0	0	4
1	100	100	0	3
1	100	100	100	2
2	100	0	0	4
2	100	100	0	3
2	100	1000	100	2

4 Tracking the Time Evolution of the Disease - Experiments and Results

In order to evaluate the efficiency of the color set back-projection algorithm in tracking the time evolution of the disease, it was tested on a gastroenterologic imagistic database at Filantropia University Hospital of Craiova. A double blind, double dummy, placebo controlled study was performed in order to evaluate the efficacy and safety of a new proton pump inhibitor in the treatment of peptic ulcer. Two hundred and two patients were included and allocated in two groups: a placebo group (102) and a treatment group (100). One endoscopic image was taken at inclusion for each patient, in order to evaluate the cicatrisation process under treatment. During the 6 weeks of follow-up, another endoscopic image of the ulcer was taken after the first 3 weeks and another one at the end of the treatment (6 weeks). Both images were stored in the database.

The endpoint of the study was to estimate the percentage of patients in which the ulcer size diminished with more than 50% after three weeks of treatment and with more than 80% at the end of the treatment. A randomization number was allocated to each patient. Each image stored at randomization was labeled XXXT0, in which XXX was the randomization number and T_0 designed the initial aspect of the ulcer, before treatment. Using the same algorithm, the image stored at the second visit (three weeks of treatment) was labeled XXXT1 and XXXT2 at the end of the treatment. Only the endoscopist made the initial evaluation, i.e. the diagnosis of peptic ulcer at T_0. In order to evaluate the colour set back-projection algorithm, at the following visits both the endoscopist and the computer performed the search in the database, and an independent expert validated the results. The validated results were quantified as „Identified" or "Non-Identified" both for the observational-based method and the computer-based one and included in statistics. The duration of the retrieval process

was also taken into account and included in the statistics. In figure 4 are presented the important regions detected by the algorithm at the moment T_0 and T_1 for the selected image. At the moment T_1, after a period of treatment the area of the region is smaller.

The percentage of correct diagnosis attended by using the classic observational method is compared with the results obtained with the computer-based one in terms of reliability and reproducibility of this method. Thus it is evaluated the advantage of using a computer-based retrieval system in terms of reducing the time spent on such an operation. For this purpose a statistical computation was used, performing the non-parametric two-sample Wilcoxon test for independent data and the analysis of variance.

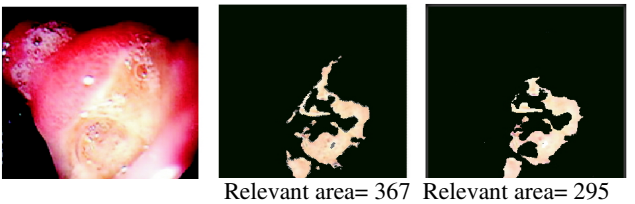


Fig. 4. Image with relevant color regions at the moment T_0 and T_1

In all, both the observer and the computer software correctly identified the healing staging at T_1 (three weeks visit) in 170 patients, i.e. 84.15% while the attended statistical value was 168 patients, and in 158 patients at T_2 (end-of-treatment visit) i.e. 78.21% while the attended statistical value was 155. At T_1, the human observer correctly staged the ulcer in 188 patients (93.07%), missing 14, while the computer software staged correctly 181 (89.60%) missing 21. At the last visit (T_2) the human observer staged correctly 186 patients (92.07%) missing 16 and the computer provided a good staging in 169 (83.66%) missing 33.

There were slight differences between human observer and computer system in the appreciating the healing staging in the way that the endoscopist missed the correct staging in certain cases and the software in others, but in all the inter-observer agreement was 84.15% at T_1 and 78.21% at T_2 as seen in table 2.

Table 2. Evaluation of inter-observer agreement in human- and computer-based query in the database

Patients: 202		Correctly staged		Missed by both computer and observer		Missed by computer but not the observer		Missed by observer but not the computer	
		Nr	%	Nr	%	Nr	%	Nr	%
T1	HO	188	93.0	3	1.48	11	5.44	8	3.96
	CS	181	89.6	3	1.48	15	7.42	18	8.91
	Both	170	84.1	3	1.48	18	8.91	11	5.44
T2	HO	186	92.0	5	2.47	11	5.44	6	2.97
	CS	169	83.6	5	2.47	23	11.38	28	13.86
	Both	158	78.2	5	2.47	28	13.86	11	5.44

Despite this, no significant statistical difference between human observer and computer system was noticed by using the Chi-square test and the Wilcoxon nonparametric test, arguing for a good liability of the method (table 3).

Table 3. Statistic analysis of differences between human- and computer-based querying in the database

	Observer-based evaluation of staging vs. computer-based evaluation at T1 (3 weeks)	Observer-based evaluation of staging vs. computer-based evaluation at T2 (end-of-treatment)
N	202	202
Missing	0	0
DOL	1	1
Chi2	1.966	2.828
p (Chi2)	0.1609	0.0927
G-square	1.589	2.396
p (G-square)	0.2075	0.1217
Contigence	0.098	0.117
Phi	0.099	0.118
corrected Chi2	0.899	1.767
corrected p	0.341	0.1838
Fisher's exact test	0.1661	0.1482
Mann-Whitney U	19695.00	18685.00
U'	21109.00	22119.00
Wilcoxon T	40198.00	39188.00
Significance level for one-sided test	11.11%	0.52%
Significance level for two-sided test	22.22%	1.03%

Table 4. Descriptive statistics of time spent by human observer and computer software to retrieve and evaluate a patient's record

Time for retrieval (seconds)	Human observer, visit T1	Computer software, T1	Human observer, visit T2	Computer software, T2
N	202	202	202	202
Mean	37.520	22.149	37.807	18.881
Std. Deviation	16.962	8.878	14.499	9.914
Std. Error	1.193	0.625	1.020	0.698
Min.	9.000	5.000	6.000	7.000
Max	76.000	50.000	76.000	54.000
Missing	0	0	0	0

The speed of the retrieval process was also tested, comparing the time spent by the observer and the computer to find each patient's record in the database. This process was electronically measured and stored in the computer for statistics. The result is that the software has a significantly higher speed than human observer with no significant

decrease of the retrieval quality. In average, at T_1 visit, the observer spent 37.52 seconds for each patient to find and evaluate the record, while the computer needed only 22.14. At visit T_2 the human used 37.80 seconds to retrieve and evaluate the record, and the computer only 18.88 seconds (table 4). These differences had statistical significance as showed the t and Z test with a 95% confidence interval (table 5).

Table 5. Evaluation of statistical significance of differences observed in terms of retrieval speed between human observer and computer software

Timing comparisons: human observer vs. computer software	Mean difference	DOL	t	p	95% inf.	95%sup
T1	15.371	201	11.937	<0.0001	12.832	17.910
T2	18.926	201	15.094	<0.0001	16.453	21.398

5 Conclusions

This article presents the experimental results obtained in two medical applications based on automated image segmentation realized with the help of the color set back-projection algorithm. Our goal was to evaluate this algorithm for detection of diseased tissue regions, since in previous work we had observed satisfactory results for region detection in nature images. We hypothesized that the endoscopic abnormalities are detectable in these images by color change.

In the first application we used these regions to query our database by image content, and inspected the retrieved images for similarity to the query image. We obtained an average of 3 relevant images in the first 5 returned by the system, when we used color and area as search parameters. The results obtained are promising, but a larger and more extensive investigation is required to properly evaluate this method for practical medical use. A more careful evaluation is needed in order to establish correctly the algorithms quality and the usefulness of the application in clinical environment. Currently, our software tool for automatic image segmentation on color and content-based region query is used in medical teaching.

Concerning the second original application, endoscopic measurement of lesions is of great importance in the design and performance of clinical trials, as, in this case of peptic ulcer disease. Although endoscopes are constructed with wide-angle lenses that significantly distort the image by creating a relative compression of points in its periphery, the presented computerized method showed a remarkable capacity to retrieve similar images in a quite large database giving an acceptable reproducibility for independent evaluations. Some authors [7] have already developed a computer program to correct the distortion of the wide-angle lens, which permits reducing errors to 1.8% +/- 2.2% ($p < 0.05$). On the other hand, evaluations of ulcer healing by measuring video-endoscopy images as a dynamic process could not be realized up to now. The documentation of the dynamic healing process shattered either on the patient's compliance or on the inconstancy of the image cut due to wobbling [8]. In

this work, instead of presenting a continuous film, instant takes of ulcer healing were processed. The inter-observer disagreement between human observer and the computer software was significantly low and the speed of the computerized method was higher, proving benefits in terms of time and cost efficiency. In the future, the computerized method will be developed in order to increase the capability of the system to recognize more complex lesions than ulcers and to diminish the percentage of errors.

References

1. Stanescu, L., Burdescu, D., Mocanu, M.: Detecting Color Regions and Content-based Region Query in databases with Medical Images. Periodica Politecnica, Transactions on Automatic Control and Computer Science, Timisoara 49(63) (2004)
2. Müller, H., et al.: Comparing feature sets for content-based medical information retrieval. In: SPIE Medical Imaging, San Diego, CA, USA (2004)
3. Müller, H., et al.: A review of content-based image retrieval systems in medicine-clinical benefits and future directions. International Journal of Medical Informatics 73, 1–23 (2004)
4. Lehmann, T.M., et al.: The IRMA Reference Database and Its Use for Content-Based Image Retrieval in Medical Applications. In: Ammenwerth, E., Gaus, W., Haux, R., Lovis, C., Pfeiffer, K.P, Tilg, B., Wichmann, H.E. (eds.) GMDS 2004 - Kooperative Versorgung - Vernetzte Forschung - Ubiquitäre Information, pp. 251–253. Verlag videel OHG, Niebüll (2004)
5. Shyu, C., et al.: ASSERT, A physician-in-the-loop content-based image retrieval system for HRCT image databases. Computer Vision and Image Understanding 75(1/2), 111–132 (1999)
6. Smith, J.R.: Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis. Ph.D. thesis Graduate School of Arts and Sciences, Columbia University (1997)
7. Vakil, N., Smith, W., Bourgeois, K.: Endoscopic measurement of lesion size: improved accuracy with image processing. Gastrointest Endosc. 40(6), 779–780 (1994)
8. Jaspersen, D., Keerl, R., Weber, R., Huppmann, A., Hammar, C.H., Draf, W.: Dynamic imaging of gastric ulcer healing using the most modern Morph-Software. Bildgebung 63(2), 101–104 (1996)
9. Henning, M., Samuel, M., Gilles, C., Pierre-Alexandre, P., Christian, L., Antoine, G.: Automatic abnormal region detection in lung CT images for visual retrieval. Swiss Medical Informatics 57, 2–6 (2005)
10. Mezaris, V., et al.: A Test-Bed for Region-Based Image Retrieval Using Multiple Segmentation Algorithms and the MPEG-7 eXperimentation Model: The Schema Reference System. In: Enser, P.G.B., Kompatsiaris, Y., O'Connor, N.E., Smeaton, A.F., Smeulders, A.W.M. (eds.) CIVR 2004. LNCS, vol. 3115, pp. 3–540. Springer, Heidelberg (2004)
11. Gordon, S., Zimmerman, G., Greenspan, H.: Image segmentation of uterine cervix images for indexing in PACS. In: CBMS 2004. Proceedings of the 17th IEEE Symposium on Computer-Based Medical Systems, Bethesda, MD (2004)
12. Burdescu, D.D., Stanescu, L.: A New Algorithm For Content-Based Region Query In Databases With Medical Images. Studies In Health Technology and Informatics. In: Medical and care Compunetics 2, vol. 114, pp. 132–139. IOS Press, Amsterdam (2005)

Hierarchical Program Representation for Program Element Matching

Fernando Berzal, Juan-Carlos Cubero, and Aída Jiménez

Dept. Computer Science and Artificial Intelligence,
ETSIIT - University of Granada, 18071, Granada, Spain
{fberzal,jc.cubero,aidajm}@decsai.ugr.es

Abstract. Many intermediate program representations are used by compilers and other software development tools. In this paper, we propose a novel representation technique that, unlike those commonly used by compilers, has been explicitly designed for facilitating program element matching, a task at the heart of many software mining problems.

1 Introduction

Program element matching is a common problem that must be addressed in many software mining applications. It is required for maintaining several versions of the same program (a.k.a. multi-version program analysis [1]), merging, regression testing automation, understanding the evolution of software code and the nature of software changes [2], detecting duplicated code (or near duplicates) for refactoring (or even bug fixing), and also for concept analysis [3], reverse engineering, and re-engineering.

Quite often, matching is approximated by comparing the textual similarity of program elements, at their source code level [4] [5]. As an alternative approach, some techniques match elements at their syntactic level [6] [7] [8] [9] [10] [11]. This enables them to detect some simple transformations that might go unnoticed at the source code level.

Even though no automatic tool can be perfectly accurate in determining the semantic equivalence of two programs (because of the inherent undecidability of the semantic program equivalence problem), this paper introduces a novel hierarchical program representation that can be useful for matching program elements in situations where existing techniques fall short.

2 Beyond Syntax Trees

A program textual description (i.e. its source code), or its equivalent syntactic representations (e.g. syntax trees and control flow graphs), describes the processes that the program has been designed to perform, at least in imperative programming languages. Substituting a state description for a process description, however, can help simplify its description, provided that we find the right representation for the program structure.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 467–476, 2007.

The program dependence graph (PDG) [12], for instance, is an intermediate program representation that makes explicit both data and control dependences. Control dependences are derived from the control flow graph and represent the essential control flow relationships in a program. Data dependences (hazards in the hardware jargon) refer to situations where instructions use data involved in the execution of preceding instructions. True data dependences, also known as RAW (read-after-write) hazards, refer to situations where a statement needs, as an operand, the result computed by a preceding statement. Dependence analysis, thus, is used to discover execution-order constraints in a software program. These constraints help determine if it is safe or not to reorder or parallelize statements, hence their importance in optimizing compilers [13] [14].

Program dependence graphs have also been used for program element matching. Unfortunately, the proposed algorithms are not directly applicable to current programming languages: they work only on limited languages without global variables, pointers, arrays, or procedures [1]; or they are just too inefficient to be of practical use.

- Horwitz [15] determines which program elements have changed by comparing two versions of a program. She builds a program representation graph (PRG) that combines features of program dependence graphs (PDG) and static single assignment forms (SSA). Then, program elements are partitioned into sets of equivalent behavior using an efficient graph partitioning algorithm. The proposed technique is able to flag *semantic* changes that might go unnoticed if we used a text-based program comparator (e.g. direct or indirect uses of changed variable values), but only in a limited language with scalar variables, assignment statements, structured control statements, and output statements.
- Krinke's approach [16] identifies similar code in programs by finding maximal isomorphic subgraphs in program dependence graphs. His algorithm detects subgraphs with identical k-length paths in attributed directed graphs. Unfortunately, high amounts of duplicated code cause exploding running times since testing graph isomorphism is NP-complete. Moreover, large duplicated code sections cause many overlapping duplicates to be reported. This is a common problem with many existing techniques that have grown out of work on compiler optimization, which requires semantic-preserving transformations (i.e. they always err on the side of flagging spurious differences and never miss a real difference).
- Jackson and Ladd's *semantic diff* [17] takes two versions of a procedure and generates a report summarizing the differences between them in terms of the observable input-output behavior of the procedure. Unlike the aforementioned approach, this tool does not check for dependence graph isomorphism. It also sacrifices soundness in the presence of aliasing and pointers. Common semantic-preserving transformations will be correctly interpreted as such (e.g. local variable renaming, using temporary variables for common subexpressions...), but some real differences will be missed (e.g. off-by-one errors). Surprisingly, the lack of alias analysis in this tool turned out not

to be a serious problem in the reported experiments since "aliases occur relatively infrequently" (sic) in practice.

– *Dex (Difference Extractor)* [2] does not use PDGs but abstract semantic graphs (ASGs) in order to compare different versions of a C program. ASGs are no more than standard abstract syntax trees with additional edges indicating type information. As a matter of fact, Dex works on ordered, rooted trees it extracts from the ASGs. Since Dex matches trees, and not graphs, it has a polynomial worst-case time complexity (it avoids working on arbitrary graphs, which would lead to NP complexity). Dex has been used for the analysis of bug fixes, but it should be extended to detect changes that involve dependences between non-contiguous program elements.

By taking dependences into account, three of the aforementioned techniques consider not only the syntactic structure of programs, but also the data flow within them. This makes them robust with respect to the relative ordering of the independent statements in a program, a feature they share with the novel program representation we now introduce.

3 Program Dependence Higraphs

In Nature, complexity frequently takes the form of hierarchic systems – the complex system being composed of subsystems that in turn have their own subsystems –, maybe because hierarchic systems can evolve far more quickly than non-hierarchic systems of comparable size [18]. Hierarchies also appear in human problem solving and, as we are all familiar with, in Software Engineering (a domain with no obvious connections with natural evolution).

In the form of trees, hierarchies provide relatively simple descriptions for complex systems when they are decomposable (or near decomposable). The novel representation model we propose, the program dependence higraph or PDH for short, lends itself to the hierarchical interpretation of any software system.

Figure 1 includes a simple code snippet whose dependence higraph is also shown. We will use this example to illustrate the construction of dependence higraphs. But before we formally present them, we must introduce some preliminary definitions.

3.1 Reduced Blocks

As in any dependence-based representation model, we first perform some preliminary control-flow analysis. However, instead of partitioning the intermediate code into *basic blocks* (i.e. maximal sequences of consecutive instructions), as any self-respecting compiler back end would do, we use *reduced blocks*, which are defined as follows:

A **reduced block** is a minimal, stand-alone, consecutive sequence of instructions whose execution *(a)* has an observable effect in the program state, or *(b)* may affect the program control flow.

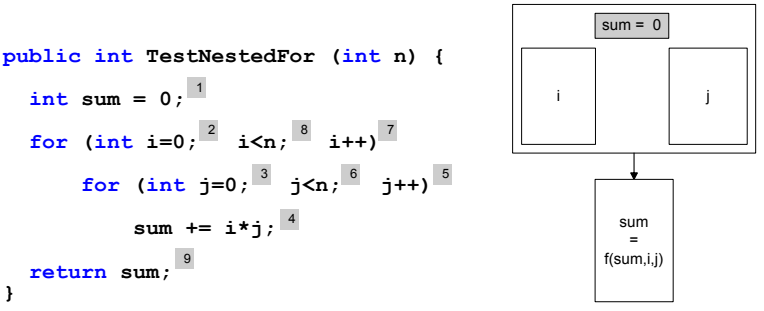


Fig. 1. Simple code snippet (left) and its corresponding dependence higraph (right)

The first case corresponds to sequences of instructions used to evaluate expressions whose results are stored into program variables (including local variables and globally accessible data). This case also applies to procedure calls with visible side effects.

The second case matches those code fragments that appear at the end of basic blocks and determine control flow. Therefore, every basic block will contain, at least, one reduced block (but may contain many of them).

If we view a basic block as a directed acyclic graph (DAG), where common subexpressions are shared among different expressions, then a reduced block is obtained from each tree derived from the DAG representation of the basic block (i.e. we explicitly introduce redundant expressions). This unusual transformation, from the compiler point of view, is intended to convert reduced blocks into small independent black boxes. In some sense, this is similar to the input-output dependence tracking found in *semantic diff* [17]).

In the example shown in Figure 1, we can identify nine reduced blocks. The numbers annotating the source code in Figure 1 mark these blocks. A control flow graph can then be derived from the set of reduced blocks. Figure 2(left) represents the control flow graph derived from our code snippet. It should be noted that, for instance, reduced blocks 4 and 5 would have been merged into a single node if we had used basic blocks instead of reduced blocks for our control-flow analysis.

3.2 The Dominance DAG

Once we have analyzed the program control flow, we must perform some data-flow analysis before we can construct the dependence higraph.

For each reduced block B_i, we define $def(B_i)$ as the set of variables whose values might be modified by the execution of B_i (this includes any variable within the current program element scope: local variables, globally-accessible data, procedure parameters, and function return values). Similarly, we define $use(B_i)$ as the set of variables whose values might be used during the execution of B_i.

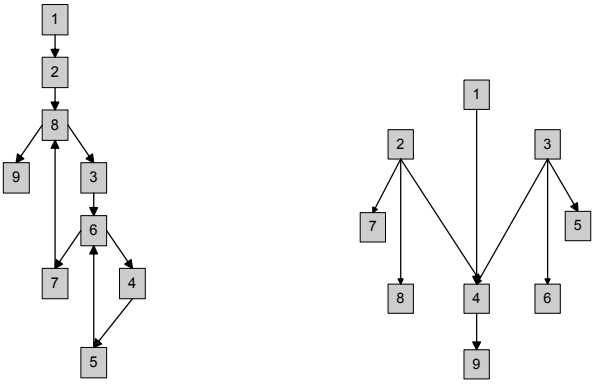


Fig. 2. The control flow graph (left) and the resulting dominance DAG (right) derived from the code snippet in Figure 1

We say that there is a *def-use chain* from B_i to B_j if a variable v defined by B_i is used by B_j (i.e. $def(B_i) \cap use(B_j) \neq \varnothing$) and there exists a path from B_i to B_j in the control flow graph where the value of v is not changed by any node in the path preceding B_j.

Now, we can define a **strong dominance relationship** as follows: A block B_i strongly dominates a block B_j if, and only if, there is at least one def-use chain from B_i to B_j *and* every path in the control flow graph from the entry node to B_j includes B_i

The dominance relationship computation can be expressed as a data-flow problem and solved using standard data-flow analysis techniques [19]. As an antisymmetric relationship defined among the reduced blocks in a program, it defines a directed acyclic graph that will serve as the basis for the construction of the program dependence higraph.

Figure 2(right) shows the dominance DAG corresponding to the code snippet in Figure 1. It should be noted that, unlike dominator trees [19], which are exclusively defined in terms of the control flow graph, dominance DAGs also incorporate data-flow information. This important difference explains why the strong dominance relationship yields directed acyclic graphs and not just trees.

3.3 Dependence Higraph Definition

Higraphs, as a general kind of diagram, are useful for displaying topological structures [20]. They have applications in databases, knowledge representation, and the behavioral specification of complex concurrent systems (e.g. Harel's statecharts and their descendants, including UML state machine diagrams).

A higraph is a graph whose nodes may contain higraphs within them. Given a higraph node n, the children of n are the nodes in the graph directly within n.

In order to define the program dependence higraph, we extend the strong dominance relationship we defined in the previous section in terms of blocks. Given

two children of a node in the higraph, n_i and n_j, we will say that n_i strongly dominates n_j if two reduced blocks B_i and B_j exist so that B_i is contained within n_i, B_j is contained within n_j, and B_i strongly dominates B_j.

A program dependence higraph (PDH), if we ignore the control and data dependences among the reduced blocks in the program, can then be defined as a tree with two kinds of nodes: P-nodes and S-nodes.

- **P-nodes** have children that are not related by the strong dominance relationship. In a very limited sense, the nodes within a P-node might be *parallelized*.
- On the other hand, **S-nodes**' children are *sequentially* connected by the strong dominance relationship. In other words, if an S-node contains children n_1, n_2... n_k, then n_i strongly dominates n_{i+1} for all the S-node children but n_k (i.e. the last node in the S-node does not dominate any other node in the S-node).

With the program dependence higraph defined as above, the reduced blocks in the program control flow graph will be, therefore, the leaves in the tree defined by the S-node and P-node containment hierarchy.

Figure 1 included the program dependence higraph corresponding to the code snippet shown at its left. The sample program is represented by an S-node whose second child, which corresponds to the `sum` value computation, strongly depends on its first child. This, in turn, is a P-node containing three independent higraphs, which roughly correspond to the `sum` value initialization and the counters that control the execution of the two nested (but independent) loops.

In the next section, we will show how we can obtain such an *intuitive* higraph from the program dependence DAG in Figure 2.

3.4 Dependence Higraph Construction

Program higraph construction can be performed by an iterative bottom-up algorithm that traverses the edges in the program dependence DAG. The program higraph construction algorithm proceeds by merging nodes until there is only one node left, which will be the root of the resulting higraph.

The traversal of the dependence DAG is done backwards in order to lump together nodes that share the same set of predecessors in the DAG:

- We create *S-nodes* containing sequences of nodes (n_1, n_2... n_k) whenever n_i is the only predecessor of n_{i+1} in the current program dependence DAG *and* either *(a)* n_i has no other successors in the current dependence DAG, or *(b)* n_{i+1} has no successors and does not share its set of predecessors with any other node in the DAG. From an intuitive point of view, S-nodes cluster sequences that are tightly related by the strong dominance relationship.
- Next, we use a greedy algorithm to discover *P-nodes* containing sets of nodes $\{n_1, n_2 ... n_k\}$ that share their sets of predecessors and successors in the current program dependence DAG. Each of this sets will lead to a new P-node in the program dependence higraph.

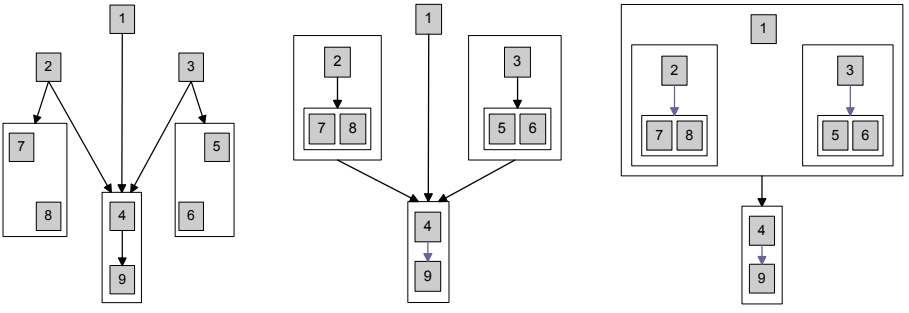


Fig. 3. Dependence higraph construction (from left to right): *(a)* initial S-node and P-nodes; *(b)* identification of S-nodes; *(c)* P-node identification

Once we create an S-node or a P-node, we replace the enclosed nodes in the dependence DAG by the newly-created node (and its corresponding strong dependences). The process is repeated until the dependence DAG is reduced to a single node, which will become the root of the program dependence higraph. Obviously, this root might be either an S-node or a P-node, depending on the particular situation.

This polynomial-time iterative algorithm eventually terminates because, starting from the finite dependence DAG derived from an imperative program, each algorithm iteration reduces the number of nodes in the DAG.

It is important to note that the two steps above must be performed in the specified order (S-nodes before P-nodes) in order to facilitate the semantic interpretation of the resulting higraph.

Let us now return to the code snippet from Figure 1, whose dependence DAG was shown in Figure 2.

In the first iteration of the PDH construction algorithm, we detect that blocks 4 and 9 can be lumped together into an S-node. Once these blocks are merged into the S-node 49, we also detect that we can create two new P-nodes, merging blocks 7 and 8 on the left side, and blocks 5 and 6 on the right side. We can do this because each pair of blocks share the same set of predecessors in the DAG (i.e. block 2 for the P-node 78, block 3 for the P-node 56). The resulting DAG is shown in Figure 3(a).

Now, the newly-created P-blocks 78 and 56 have only one predecessor, have no successors, and they do not share their sets of predecessors with any other nodes in the current higraph. Therefore, two new S-nodes are created as shown in Figure 3(b).

Next, by a backwards traversal of the current DAG, we find that nodes 278, 1, and 356 share their predecessors (the empty set, since none of them has any predecessor) and they also share their only successor. Hence, we lump them into a new P-node and our original DAG has been reduced to a simple two node higraph, as shown in Figure 3(c).

Therefore, our program dependence higraph consists of an S-node with two children: an initial P-node and a nested S-node following the initial P-node. The resulting higraph matches the one we introduced in Figure 1 and the reader can now check the semantic interpretation we provided when we introduced the idea of dependence higraphs in the previous section.

4 Dependence Higraph Matching Abilities

The main limitation of text- and syntax-based matching algorithms is that relatively simple modifications can hinder the detection of code duplication. A matching tool based on dependences is not so easily confused. We can analyze the potential matching abilities each approach provides by studying some common change scenarios:

- **Verbatim copying** (as in copy & paste programming), for which any technique should properly work.
- **Text insertion/deletion** (adding or removing source code comments, changing I/O messages, and other minor formatting modifications) might reveal some limitations of naive string matching algorithms, although every clone detector should still handle this kind of changes.
- **Renaming** (changing the names of program elements: variables; functions, procedures, or methods; classes, modules, or packages): A clone detector provided with a string tokenizer can handle such changes.
- **Altering assignments and expressions** (adding, removing, or changing variables and expressions in assignment statements): Most clone detectors would also detect this change scenario.
- **Control flow modifications** (changing the circumstances the code will be executed under; e.g. insertion of new conditional statements or changing existing conditions in conditional expressions): A cleverly-devised text-based matching algorithm would detect them, although some syntax-based clone detectors can be misled.
- **Replacement** (changing expressions and control statements for equivalent ones, e.g. `for` → `while`): Most text-based and some syntax-based matching algorithms will fail. Dependence-based algorithm will properly work.
- **Reordering** (changing the order of independent statements while preserving the program semantics, e.g. exchanging `for` loops in the example from Figure 1): In this situation, only dependence-based techniques are reliable.
- **Splitting** (splitting a procedure or module into several ones, as in the 'method extraction' refactoring) and **Merging** (merging procedure bodies, as in procedure inlining): Only dependence higraphs can be truly useful here, since they can benefit from existing embedded tree mining algorithms [21].

As Table 1 shows, syntax-based techniques generally improve token-based matching, even though they might fail to properly match code blocks within new

Table 1. The potential matching abilities of alternative program representation techniques (■ excellent, ⊠ partial, □ poor)

	String matching	**Lexical analysis**	**Syntactic analysis**	**Dependence graphs**	**Dependence higraphs**
Verbatim copying	■	■	■	■	■
Text insertion/deletion	⊠	■	■	■	■
Renaming	□	⊠	■	■	■
Assignment modifications	□	■	■	■	■
Control flow alteration	□	■	⊠	■	■
Replacement	□	⊠	⊠	■	■
Reordering	□	□	⊠	■	■
Splitting	□	□	□	⊠	■
Merging	□	□	□	⊠	■

conditional statements. Dependence-based techniques are robust with respect to this kind of modifications and they are also better at the detection of equivalent control structures. Finally, they are not easily confounded by semantic-preserving statement reordering.

5 Conclusions and Future Work

Text matching tools are usually line-based, so even lexical changes with no syntactic effects might spuriously appear as modifications. Syntactic matching tools are able to properly detect some code transformations, but they are only able to show the syntactic scope of the detected change. This could be a severe drawback if we are interested in understanding the effects of a given change (as maintainers must do not to introduce new bugs when modifying an existing code base).

Semantic matching techniques, and the use of dependence graphs in particular, eliminate this limitation, since the repercussions of a change can be determined by following dependence edges in the dependence graph. Unfortunately, traditional semantic-based techniques are computationally expensive (global analysis is often unfeasible in large systems).

Dependence higraphs, however, are amenable to some optimizations due to their hierarchical structure. S-nodes, due to their sequential structure, can be matched as ordered, rooted trees to find perfect matches (dynamic programming can be used for approximate matches). Unordered P-nodes matching can be sped up with the help of heuristics.

Another interesting (and differentiating) feature of dependence higraphs is that, due to their hierarchical nature, they are well suited for detecting some change scenarios beyond the scope of previous techniques (procedure splitting and merging, for instance). This fact opens up new possibilities in the study of refactorings, aspect mining, and software modularization in general.

References

1. Kim, M., Notkin, D.: Program element matching for multi-version program analyses. In: MSR 2006. Proceedings of the 2006 International Workshop on Mining Software Repositories, pp. 58–64 (2006)
2. Raghavan, S., Rohana, R., Leon, D., Podgurski, A., Augustine, V.: Dex: A semantic-graph differencing tool for studying changes in large code bases. In: ICSM 2004, pp. 188–197 (2004)
3. Tonella, P.: Formal concept analysis in software engineering. In: ICSE 2004, pp. 743–744 (2004)
4. Hunt, J.W., Szymanski, T.G.: A fast algorithm for computing longest common subsequences. Communications of the ACM 20(5), 350–353 (1977)
5. Tichy, W.F.: The string-to-string correction problem with block moves. ACM Trans. Comput. Syst. 2(4), 309–321 (1984)
6. Yang, W.: Identifying syntactic differences between two programs. Software – Practice & Experience 21(7), 739–755 (1991)
7. Hunt, J.J., Tichy, W.F.: Extensible language-aware merging. In: ICSM 2002, pp. 511–520 (2002)
8. Neamtiu, I., Foster, J.S., Hicks, M.: Understanding source code evolution using abstract syntax tree matching. In: MSR 2005. Proceedings of the 2005 International Workshop on Mining Software Repositories, pp. 282–290 (2005)
9. Laski, J., Szermer, W.: Identification of program modifications and its applications insoftware maintenance. In: ICSM 1992, pp. 1–5 (1992)
10. Apiwattanapong, T., Orso, A., Harrold, M.J.: A differencing algorithm for object-oriented programs. In: ASE 2004, pp. 2–13 (2004)
11. Wang, Z., Pierce, K., McFarling, S.: Bmat – a binary matching tools for stale profile propagation. Journal of Instruction-Level Parallelism 2 (2000)
12. Ferrante, J., Ottenstein, K.J., Warren, J.D.: The program dependence graph and its use in optimization. ACM Trans. Program. Lang. Syst. 9(3), 319–349 (1987)
13. Muchnick, S.: Advanced Compiler Design and Implementation. Morgan Kaufmann, San Francisco (1997)
14. Allen, R., Kennedy, K.: Optimizing Compilers for Modern Architectures: A Dependence-based Approach. Morgan Kaufmann, San Francisco (2001)
15. Horwitz, S.: Identifying the semantic and textual differences between two versions of a program. In: PLDI 1990, pp. 234–245 (1990)
16. Krinke, J.: Identifying similar code with program dependence graphs. In: WCRE 2001, pp. 301–309 (2001)
17. Jackson, D., Ladd, D.A.: Semantic diff: A tool for summarizing the effects of modifications. In: ICSM 1994, pp. 243–252 (1994)
18. Simon, H.A.: The Sciences of the Artificial, 3rd edn. The MIT Press, Cambridge (1996)
19. Aho, A.V., Lam, M.S., Sethi, R., Ullman, J.D.: Compilers: Principles, Techniques, and Tools, 2nd edn. Addison-Wesley, Reading (2006)
20. Harel, D.: On visual formalisms. Communications of the ACM 31(5), 514–530 (1988)
21. Zaki, M.J.: Efficiently mining frequent embedded unordered trees. Fundamenta Informaticae 66(1-2), 33–52 (2005)

A Combination-of-Tools Method for Learning Interpretable Fuzzy Rule-Based Classifiers from Support Vector Machines

Tamas Kenesei, Johannes A. Roubos, and Janos Abonyi

Department of Process Engineering, University of Pannonia
Veszprem P.O.Box 158, H-8201 Hungary
abonyij@fmt.uni-pannon.hu
www.fmt.uni-pannon.hu/softcomp

Abstract. A new approach is proposed for the data-based identification of transparent fuzzy rule-based classifiers. It is observed that fuzzy rule-based classifiers work in a similar manner as kernel function-based support vector machines (SVMs) since both model the input space by nonlinearly maps into a feature space where the decision can be easily made. Accordingly, trained SVM can be used for the construction of fuzzy rule-based classifiers. However, the transformed SVM does not automatically result in an interpretable fuzzy model because the SVM results in a complex rule-base, where the number of rules is approximately 40-60% of the number of the training data. Hence, reduction of the SVM-initialized classifier is an essential task. For this purpose, a three-step reduction algorithm is developed based on the combination of previously published model reduction techniques. In the first step, the identification of the SVM is followed by the application of the Reduced Set method to decrease the number of kernel functions. The reduced SVM is then transformed into a fuzzy rule-based classifier. The interpretability of a fuzzy model highly depends on the distribution of the membership functions. Hence, the second reduction step is achieved by merging similar fuzzy sets based on a similarity measure. Finally, in the third step, an orthogonal least-squares method is used to reduce the number of rules and re-estimate the consequent parameters of the fuzzy rule-based classifier. The proposed approach is applied for the Wisconsin Breast Cancer, Iris and Wine classification problems to compare its performance to other methods.

Keywords: Classification, Fuzzy classifier, Support Vector Machine, Model Reduction.

1 Introduction

Fuzzy logic helps to improve the interpretability of knowledge-based classifiers through its semantics that provide insight in the classifier structure and decision making process. Application of SVM methods for fuzzy logic is not a completely new idea. SVMs based on sigmoidal kernels are functionally equivalent to feed-forward neural networks. When Gaussian kernel functions are applied, the SVM

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 477–486, 2007.

can be used to initialize Radial Basis Function (RBF) classifiers [8]. In [9] a support vector regression model was used to identify a dynamic process by a RBF network. A restricted class of fuzzy systems is functionally equivalent to RBF models [11]. These SVM-based approaches can also be applied to identify fuzzy systems as shown in [10]. However, only a simple example for the approximation of an univariate function was presented, while interpretability and applicability issues were not discussed.

An example for the incorporation of fuzzy logic into SVM can be found in [12], where fuzzy sigmoid functions has been used as a kernel in the SVM framework. An another interesting example can be found in [13], where the concept of fuzzy regression has been adopted. Another approach for the fusion of fuzzy and SVM based modeling techniques us proposed in [14], where fuzzy membership values are obtained from fuzzy clustering have been applied to form input variables of the SVM algorithm. The linguistic interpretability of the kernel functions of SVMs has been already recognized in [15]. It has been showed that the kernel matrix may be interpreted in terms of linguistic values based on the premises of if–then rules. A simple real–life application example for SVM based fuzzy modeling has been presented in [16], where the SVM based fuzzy model is used for modeling the arc welding process. Though simple SVM models can handle with a variety of classification tasks, fuzzy support vector machines have been investigated for a period of time to make classification become more effective Lin and Wang (2002) first proposed a prototype of fuzzy SVM (FSVM), where one applies a fuzzy membership function to each input data of SVM. This kind of fuzzy SVM has been applied in [17] for text categorization.

The main advantage of rule-based fuzzy rule-based classifiers over SVM is the transparency and the linguistic interpretability. Fuzzy logic, however, does not guarantee interpretability [1,3]. Hence real effort must be made to keep the resulting rule-base transparent [2,4,5]. For this purpose, two main approaches are followed in the literature: (i) selection of a low number of input variables (features) in order to create a compact classifier [6], and (ii) construction of a large set of possible rules by using all inputs, and subsequently use this set to make a useful selection out of these rules [4,7]. In both approaches, further model reduction can be realized by generalization and/or similarity-based set-reduction techniques [3,6]. However, if the fuzzy model is obtained from a SVM, the resulted initial fuzzy model is usually not very interpretable. This is because the SVM results in a complex rule-base where the number of rules is approximately 40-60% of the number of the training data. Therefore, fuzzy models constructed from SVM are presumably more complex than necessary. This suggests that a simple transformation of a SVM into a fuzzy model should be followed by rule-base simplification steps. Hence, to obtain a parsimonious and interpretable fuzzy rule-based classifier a three-step model reduction algorithm is proposed:

1. *Application of the Reduced Set method*
   The identification of the SVM is followed by the application of the Reduced Set (RS) method to decrease the number of kernel functions. Originally, this method has been introduced by [18] to reduce the computational complexity

of SVMs. The obtained SVM is subsequently transformed into a fuzzy rule-based classifier.

2. *Similarity-based fuzzy set merging*
   The Gaussian membership functions of the fuzzy rule-based classifier are derived from the Gaussian kernel functions of the SVM. The interpretability of a fuzzy model highly depends on the distribution of the membership functions. Hence, the next reduction step is achieved by merging fuzzy sets based on a similarity measure [3].
3. *Rule-base simplification by orthogonal transformations*
   Finally, an orthogonal least-squares method is used to reduce the number of rules and re-estimate the consequent parameters of the classifier. The application of orthogonal transforms for reducing the number of rules has received much attention in the recent literature [19,20]. These methods evaluate the output contribution of the rules to obtain the order of importance. The less important rules are then removed according this ranking to further reduce the complexity and increase the transparency.

The remainder of this article is organized as follows. Section 2 explains the structure of the fuzzy rule-based classifier. In Section 3, the reduction of the classifier is discussed. First, the SVM learning method is reviewed. The other subsections deal with the similarity-based and orthogonal transform-based model simplification techniques. In Section 4, the proposed approach is experimentally evaluated for the two-class Wisconsin Breast Cancer classification problem and the three-class Iris and Wine classification problems. Finally, the conclusions are given in Section 5.

2 The Fuzzy Rule-Based Classifier

2.1 The Structure of the Fuzzy Rule-Based Classifier

Fuzzy rule-based models have been successfully applied to many pattern recognition and classification problems. The classical fuzzy rule-based classifier classifies an example according to the rule with the greatest degree of association. By using this reasoning method the information provided by the other rules is lost. In this paper a new fuzzy rule-based classifier structure is presented for the classification of N_c labeled classes.

One widely used approach to solve non-fuzzy N_c-class pattern recognition problems is to consider the general problem as a collection of binary classification problems. Accordingly, N_c classifiers can be constructed, i.e. one for each class. The c–th classifier, $c = 1, \ldots, N_c$, separates class c from the $N_c - 1$ other classes. This one-against-all method results in a hierarchical classifier structure that allows for a sequential model construction and evaluation procedure. Based on this, we propose fuzzy rule-based classifier that consists of N_c fuzzy subsystems with a set of Takagi-Sugeno-type fuzzy rules [21] that describe the c–th class in the given data set as

$$R_i^c: \quad \textbf{If } x_1 \text{ is } A_{i1}^c \textbf{ and } \ldots x_n \text{ is } A_{in}^c \textbf{ then } y_i^c = \delta_i^c, \quad i = 1, \ldots, N_R^c\,, \quad (1)$$

where R_i^c is the i–th rule in the c–th fuzzy rule-based classifier and N_R^c denotes the number of rules. $A_{i1}^c, \ldots, A_{N_i}^c$ denote the antecedent fuzzy sets that define the operating region of the rule in the N_i-dimensional input space; $\mathbf{x} = [x_1, x_2, \ldots, x_{N_i}]^T$. The rule-consequent δ_i^c, $i = 1, \ldots, N_R^c$, is a crisp (non-fuzzy) number. The **and** connective is modeled by the product operator allowing for interaction between the propositions in the antecedent. Hence, the degree of activation of the i–th rule is calculated as $beta_i^c(\mathbf{x}) = \prod_{j=1}^{N_i} A_{ij}^c(x_j), \;\; i = 1, \ldots, N_R^c$.

The output of the classifier is determined by the following decision function

$$y^c = sgn\left(\sum_{i=1}^{N_R^c} \beta_i^c(\mathbf{x})\delta_i^c + b^c\right), \tag{2}$$

where b^c is a constant threshold. If $y^c = +1$ then the observation $\mathbf{x}$ is part of class c, otherwise, when $y^c = -1$, then it is not an item in class c.

For simplicity in the notation, the superscript c that denotes the index of the fuzzy subsystem is neglected in the sequel. It is not necessary there because the following sections only deal with the identification of the submodels that are independent from each other.

2.2 Formulation of the Fuzzy Rule-Based Classifier as a Kernel Machine

Recent years have witnessed a surge of interest in learning methods based on Mercer kernels, i.e. functions $k(\mathbf{x}_i, \mathbf{x}_j)$ which for all data pairs $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N_d}\} \subset \mathbb{R}^{N_i}$ give rise to positive matrices $K_{ij} := k(\mathbf{x}_i, \mathbf{x}_j)$, where N_d denotes the number of data points. Using k instead of a dot product in $\mathbb{R}^{N_i}$ corresponds to mapping the data into a possibly high-dimensional space F, by a usually nonlinear map $\phi : \mathbb{R}^{N_i} \to F$, and taking the dot product there $k(\mathbf{z}_i, \mathbf{x}) = (\phi(\mathbf{z}_i) \cdot \phi(\mathbf{x}))$ [18].

The main principle of kernel-based support vector classifiers is the identification of a linear decision boundary in this high-dimensional feature-space.A link to the fuzzy model structure will now be established. The fuzzy sets are represented in this paper by Gaussian membership functions

$$A_{ij}(x_j) = \exp\left(\frac{(x_j - z_{ij})^2}{2\sigma^2}\right) \;\; \beta_i(\mathbf{x}) = k(\mathbf{z}_i, \mathbf{x}) = \exp\left(\frac{\|\mathbf{x} - \mathbf{z}_i\|^2}{2\sigma^2}\right). \tag{3}$$

The degree of fulfillment $\beta_i(\mathbf{x})$ can be written in a more compact form by using the Gaussian kernels

This kernel interpretation of fuzzy systems shows that fuzzy models are effective in solving nonlinear problems because they map the original input space into a nonlinear feature space by using membership functions similar to the SVM that utilizes kernel functions for this purpose.

3 Fuzzy Rule-Based Classifier Based SVM

In the previous section, a new type of fuzzy rule-based classifier structure has been presented that is formed as a set of binary classifiers. We continue with

the identification of the separate fuzzy rule-based classifiers which means the determination of the number of rules, the parameters of the membership functions and the rule consequents. The known fuzzy identification methods do not estimate these parameters simultaneously, but apply heuristic tools like fuzzy clustering. However, this results in a suboptimal model structure, although it may also be subject to further optimization [5].

We aimed at a new identification method that quickly, at low computational costs, results in an optimal classifier that generalizes well. In this section we show how such fuzzy rule-based classifiers can be identified by means of SVMs. After a brief review on SVM learning, the model transformation and simplification techniques are discussed.

In the previous sections it has been shown how a SVM, that is structurally equivalent to a fuzzy model, can be identified. Unfortunately, this identification method cannot be used directly for the identification of interpretable fuzzy systems because the number of the support vectors is usually very large. Typical values are 40-60% of the number of training data which is in our approach equal to the number of rules in the fuzzy system. Therefore, there is a need for an interpretable approximation of the support vector expansion. For this purpose a step-wise algorithm will be introduced, where the first step is based on the recently published Reduced Set (RS) method developed for reducing the computational demand of the evaluation of SVMs [18].

3.1 Reduction of the Number of Fuzzy Sets

In the previous section, it has been shown how kernel-based classifiers with a given number of kernel functions N_R, can be obtained. Because the number of the rules in the transformed fuzzy system is identical to the number of kernels, it is extremely important to get a moderate number of kernels in order to obtain a compact fuzzy rule-based classifier.

From (3) it can be seen that the number of fuzzy sets in the identified model is $N_s = N_R N_i$. The interpretability of a fuzzy model highly depends on the distribution of these membership functions. With the simple use of (3), some of the membership functions may appear almost undistinguishable. Merging similar fuzzy sets reduces the number of linguistic terms used in the model and thereby increases the transparency of the model. This reduction is achieved by a rule-base simplification method [3,4], based on a similarity measure $S(A_{ij}, A_{kj})$, $i, k = 1, \ldots, n$ and $i \neq j$. If $S(A_{ij}, A_{kj}) = 1$, then the two membership functions A_{ij} and A_{kj} are equal. $S(A_{ij}, A_{kj})$ becomes 0 when the membership functions are non-overlapping. During the rule-base simplification procedure similar fuzzy sets are merged when their similarity exceeds a user-defined threshold $\theta \in [0, 1]$.

The set-similarity measure can be based on the set-theoretic operations of intersection and union $S(A_{ij}, A_{kj}) = \frac{|A_{ij} \cap A_{kj}|}{|A_{ij} \cup A_{kj}|}$, where $|.|$ denotes the cardinality of a set, and the $\cap$ and $\cup$ operators represent the intersection and union, respectively, or it can be based on the distance of the two fuzzy sets[3]. Here,

the following expression was used to approximate the similarity between two Gaussian fuzzy sets [4]

$$S(A_{ij}, A_{kj}) = \frac{1}{1 + d(A_{ij}, A_{kj})} = \frac{1}{1 + \sqrt{(z_{ij} - z_{kj})^2 + (\sigma_{ij} - \sigma_{kj})^2}} . \quad (4)$$

3.2 Reduction of the Number of Rules by Orthogonal Transforms

By using the previously presented SVM identification and reduction techniques, the following fuzzy rule-based classifier has been identified

$$y = sgn\left(\sum_{i=1}^{N_R}\prod_{j=1}^{N_i}\exp\left(\frac{(x_j - z_{ij})^2}{2\sigma^2}\right)\delta_i + b\right). \quad (5)$$

Because the application of the RS method and the fuzzy set merging procedure the obtained membership functions only approximate the original feature space identified by the SVM. Hence, the $\delta = [\delta_1, \ldots, \delta_{N_r}]^T$ consequent parameters of the rules have to be re-identified to minimize the difference between the decision function of the support vector machine and the fuzzy model (5)

$$MSE = \sum_{j=1}^{N_d}\left(\sum_{i=1}^{N_x}\gamma_i k(\mathbf{x}_j, \mathbf{x}_i) - \sum_{i=1}^{N_R}\delta_i\beta_i(\mathbf{x}_j)\right)^2 = \|\mathbf{y}_s - \mathbf{B}\delta\|^2 , \quad (6)$$

where the matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_{N_R}] \in \mathbb{R}N_d \times N_R$ contains the firing strength of all N_R rules for all the inputs $\mathbf{x}_i$, where $\mathbf{b}_j = [\beta_j(\mathbf{x}_1), \ldots, \beta_j(\mathbf{x}_{N_d})]^T$. As the fuzzy rule-based classifier (5) is linear in the parameters δ, (6) can be solved by a least-squares method $\delta = \mathbf{B}^+\mathbf{y}_s$, where $\mathbf{B}^+$ denotes the Moore-Penrose pseudo inverse of $\mathbf{B}$.

The application of orthogonal transforms for the above mentioned regression problem (6) for reducing the number of rules has received much attention in recent literature [19,20]. These methods evaluate the output contribution of the rules to obtain an importance ordering. For modeling purposes, the Orthogonal Least Squares (OLS) is the most appropriate tool [19]. The OLS method transforms the columns of $\mathbf{B}$ into a set of orthogonal basis vectors in order to inspect the individual contribution of each rule. To do this, Gram-Schmidt orthogonalization of $\mathbf{B} = \mathbf{WA}$ is used, where $\mathbf{W}$ is an orhogonal matrix $\mathbf{W}^T\mathbf{W} = \mathbf{I}$ and $\mathbf{A}$ is an upper triangular matrix with unity diagonal elements. If $\mathbf{w}_i$ denotes the i–th column of $\mathbf{W}$ and g_i is the corresponding element of the OLS solution vector $\mathbf{g} = \mathbf{A}\delta$, the output variance $\mathbf{y}_s^T\mathbf{y}_s/N_d$ can be explained by the regressors $\sum_{i=1}^{N_r} g_i\mathbf{w}_i^T\mathbf{w}_i/N_d$. Thus, the error reduction ratio, ϱ, due to an individual rule i can be expressed as $\varrho^i = \frac{g_i^2\mathbf{w}_i^T\mathbf{w}_i}{\mathbf{y}_s^T\mathbf{y}_s}$. This ratio offers a simple mean for ordering the rules, and can be easily used to select a subset of rules in a forward-regression manner. Evaluating only the approximation capabilities of the rules, the OLS method often assigns high importance to a set of redundant or correlated rules. To avoid this, in [20] some extension for the OLS method were proposed.

4 Application Example

In order to examine the performance of the proposed identification method, firstly it was applied to the Wisconsin Breast Cancer data (WBCD)[1], which is a benchmark problem in the classification and pattern recognition literature. The data is divided into a training and an evaluation subset that have similar size and class distributions (We used 342 cases for training and 341 cases for testing the classifier).

First, the advanced version of C4.5 was applied to obtain an estimate for the useful features. This method gave 36 misclassification (5.25%) for this problem. The constructed decision-tree model had 25 nodes and used mainly three inputs; x_1, x_2 and x_6. Based on this pre-study, only the previous three inputs were applied to identify the SVM classifier with $N_x = 71$ support vectors. The application of this model resulted in 3 and 15 misclassifications on the training and testing data, respectively. This model has been reduced by the RS method, by which we tried to reduce the model by a factor of 10, $N_R = 8$. By this step, the classification performance slightly decreased on the training set to 12 misclassifications, but the validation data showed a slightly better result with 14 misclassifications. Next, the reduced kernel-classifier was transformed into a fuzzy system. Figure 1 shows the membership functions that were obtained. The

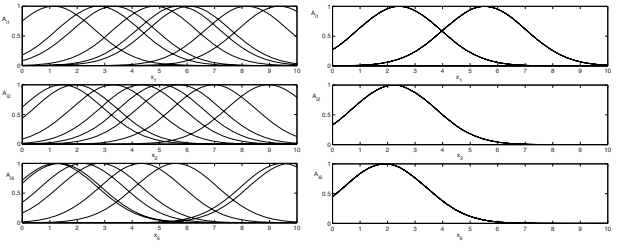


Fig. 1. Non-distinguishable membership functions obtained after the application of the RS method (left); Interpretable membership functions of the OLS reduced fuzzy model (right)

obtained model with eight rules is still not really well interpretable; however, some of the membership functions appear very similar and can probably be merged easily without loss in accuracy. The performance of the classifier slightly increased after this merging step (Table 1). Subsequently, using the OLS method, the rules were ordered according to there importance. Then, we reduced the number of rules one-by-one according the OLS ranking, till a major drop in the performance was observed. To our surprise, only two rules and four membership functions were necessary to have a good classification performance on this

[1] The WBCD is available from the University of California, Irvine, URL: *http://www.ics.uci.edu/˜mlearn/*

problem: 14 and 16 misclassification on the learning and validation data, respectively (Table 1). The obtained rules are:

R_1: **If** x_1 is Small **and** x_2 is Small **and** x_6 is Small **then** Class is Benign;
R_2: **If** x_1 is High **then** Class is Malignant;

where x_1 is the clump thickness, x_2 the uniformity of cell size, and x_6 a measure for bare nuclei.

Table 1. Classification rates and model complexity for classifiers constructed using the proposed techniques

Method	♯ Miss. Train (Accuracy)	♯ Miss. Test	♯ Rules	♯ Conditions
SVM	3 (99.1%)	15 (95.6%)	71	213
RS method	12 (96.5%)	14 (95.9%)	8	24
Membership merging	11 (96.8%)	13 (96.2%)	8	10
OLS	14 (95.9%)	16 (95.3%)	2	4

The previously presented illustrative example showed the details of how the application of the model reduction steps effect the model complexity and model performance. To give more, detailed analysis of the proposed algorithm additional benchmark datasets were also used. In the following the results of this analysis are only briefly presented. The IRIS and the Wine datasets are also taken from the UCI repository. It is interesting to note that these datasets do not define binary classification problems, hence, there was a need for the integration of three compact models identified based on the one-class from all strategy. Table 2 concludes these results. In these experiments the same parameterset was used, that points on that the proposed algorithm is robust, and its parameters can be tuned easily.

Table 2. Classification rates and model complexity for classifiers constructed using the proposed technique for the IRIS/WINE datasamples

Method	♯ Miss. Train (Accuracy)	♯ Rules	♯ Conditions
SVM	4 (97%)/5 (97%)	264/117	1056/351
RS method	8 (95%)/7 (96%)	84/63	336/189
Membership merging	12 (92%)/21 (88%)	60/27	240/81
OLS	15 (90%)/12 (93%)	36/27	54/81

5 Conclusions

It has been shown in a mathematical way that SVMs and fuzzy rule-based classifiers work in a similar manner as both models maps the input space of the classifier into a feature space with the use of either nonlinear kernel or membership functions. The main difference between SVMs and fuzzy rule-based classifier systems is that fuzzy systems have to fulfill two objectives simultaneously, i.e., they must provide a good classification performance and must also be linguistically interpretable, which is not an issue for SVMs. However, as the structure identification of fuzzy systems is a challenging task, the application of kernel-based methods for model initialization could be advantageous because of the high performance and the good generalization properties of these type of models. Accordingly, support vector-based initialization of fuzzy rule-based classifiers is proposed. First, the initial fuzzy model is derived by means of the SVM learning algorithm. Then the SVM is transformed into an initial fuzzy model that is subsequently reduced by means of the reduced set method, similarity-based fuzzy set merging, and orthogonal transform-based rule-reduction. Because these rule-base simplification steps do not utilize any nonlinear optimization tools, it is computationally cheap and easy to implement.

The application of the proposed approach is shown for the Wisconsin Breast Cancer classification problem. This classification study showed that a proper rule structure is obtained by the proposed model identification procedure. The obtained classifier is very compact but its accuracy is still comparable to the best results reported in the literature-based on nonlinear optimization tools. Besides, it might be clear that still real progress can be made in the development of novel methods for feature selection.

We consider this paper also as a case study for further developments in the direction of a combination–of–tools methodology for modelling and identification, aiming at models that perform well on multiple criteria, i.e, here different soft-computing tools, namely support vector machines and fuzzy techniques are combined to achieve a predefined trade-off between performance and transparency. In this sense, it is expected that the current work is not only useful for the identification of fuzzy classifiers but also provides insights in the understanding and analysis of SVM-based classifiers.

Acknowledgments. The authors would like to acknowledge the support of the Cooperative Research Centre (VIKKK, project III/1) and Hungarian Research Found (OTKA T049534). Janos Abonyi is grateful for the support of the Bolyai Research Fellowship of the Hungarian Academy of Sciences and the Öveges Fellowship.

References

1. de Valente Oliveira, J.: Semantic constraints for membership function optimization. IEEE Trans. FS 19, 128–138 (1999)
2. Nauck, D., Kruse, R.: Obtaining interpretable fuzzy classification rules from medical data. Artificial Intelligence in Medicine 16, 149–169 (1999)

3. Setnes, M., Babuška, R., Kaymak, U., van Nauta Lemke, H.R.: Similarity measures in fuzzy rule base simplification. IEEE Trans. SMC-B 28, 376–386 (1998)
4. Jin, Y.: Fuzzy Modeling of High-Dimensional Systems. IEEE Trans. FS 8, 212–221 (2000)
5. Setnes, M., Roubos, J.A.: GA-Fuzzy Modeling and Classification: Complexity and Performance. IEEE Trans. FS 8, 509–522 (2000)
6. Roubos, J.A., Abonyi, J., Setnes, M.: Learning fuzzy classification rules from labeled data. International Journal of Information Sciences 150, 77–93 (2003)
7. Ishibuchi, H., Nakashima, T., Murata, T.: Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. IEEE Trans. SMC–B 29, 601–618 (1999)
8. Schölkopf, B., Sung, K., Burges, C., Girosi, F., Niyogi, P., Poggio, T., Vapnik, V.: Comparing support vector machines with gaussian kernels to radial basis function classifiers. IEEE Trans Sign. Processing 45, 2758–2765 (1997)
9. Chan, W.C., Cheung, K.C., Harris, C.J.: Modelling of Nonlinear Dynamic Systems Using Support Vector Machines. In: Proc. of the IFAC Symposium on Artificial Intelligence, Budapest, Hungary, pp. 217–222 (2000)
10. Jeng, J.T., Lee, T.T.: Support Vector Machines for Fuzzy Neural Networks. In: Proc. of the IEEE SMC conference, pp. VI–115–120 (1999)
11. Jang, J.-S.R., Sun, C.-T.: Functional Equivalence Between Radial Basis Function Networks and Fuzzy Inference Systems. IEEE Trans. NN 4, 156–159 (1993)
12. Camps–Valls, G., Martn–Guerrero, J.D., Rojo-lvarez, J.L., Soria–Olivas, E.: Fuzzy sigmoid kernel for support vector classifiers. Neurocomputing 62, 501–506 (2004)
13. Yao, C.–C., Yu, P.-T.: Fuzzy regression based on asymmetric support vector machines. Applied Mathematics and Computation 182, 175–193
14. Celikyilmaz, A., Türksen, B.I.: Fuzzy Functions with Support Vector Machines, Accepted Manuscript to appear in Information Sciences
15. Leski, J.M.: On support vector regression machines with linguistic interpretation of the kernel matrix. Fuzzy Sets and Systems 157, 1092–1113 (2006)
16. Huang, X., Chen, S.: SVM–based fuzzy modeling for the arc welding process. Material Science and Engineering A 427, 181–187 (2006)
17. Wang, T.–Y., Chiang, H.-m.: Fuzzy support vector machine for multi–class text categorization. Information Processing and Management 43, 914–929 (2007)
18. Schölkopf, B., Mika, S., Burges, C.J.C., Knirsch, P., Müller, K.-R., Rätsch, G., Smola, A.: Input space vs. feature space in kernel-based methods. IEEE Trans. on Neural Networks 10(5), 1000–1017 (1999)
19. Yen, J., Wang, L.: Simplifying fuzzy rule-based models using orthogonal transformation methods. IEEE Trans. SMC-B 29, 13–24 (1999)
20. Setnes, M., Hellendoorn, H.: Orthogonal transforms for ordering and reduction of fuzzy rules. In: FUZZ-IEEE, San Antonio, Texas, USA, pp. 700–705 (2000)
21. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its application to modeling and control. IEEE Trans. SMC 15, 116–132 (1985)

An Effective Content-Based Image Retrieval System by Hierachical Segmentation*

Mingxin Zhang[1,2], Zhaogan Lu[1], and Junyi Shen[1]

[1] Dept. of Inf. and Comm. Eng., Xi'an Jiaotong University, Xi'an 710049, P.R. China
[2] Dept. of Comptuter Engineering, Lanzhou Polytechnical College, Lanzhou 730070 China
zhangmx@mail.xjtu.edu.cn, luzhaogan@mailst.xjtu.edu.cn

Abstract. As the inaccuracy of current image segmentation methods, it's unavoidable for the objects with discrepant components to be segmented into different regions. As a result, good image retrieval performance could not be achieved by those region-based image retrieval approaches. Furthermore, the complexity of image segmentations is also an unmanageable issue in the scenarios with complex backgrounds. Aspired by the wavelet multi-resolution analysis, the objects with different scales, orientations, and locations, can be retrieved by their invariant features and hierarchical multi-resolution segmentations. For simplification, the hierarchical segmentation is conducted to segment one image into equal block with different shifts and sizes in one hierarchical way, and those blocks can form a complete pitch to the image at different hierarchical levels with different shifts and sizes. The smaller the sizes of blocks are, the higher the hierarchical levels. Then, the similar metrics of these sub-blocks to query image, are evaluated to retrieve those sub-blocks with contents in query images. Meanwhile, the location and scale information about query objects can also be returned in retrieved images. With geometric invariants, normalized histograms and their combinations as invariant features, the hierarchical segmentation-based image retrieval scheme are tested by experiments via one image database with 500 images. The retrieval accuracy with geometric invariants as invariant features can achieve 78% for the optimal similar metric threshold, only inferior to that of region-based image retrieval schemes, whose retrieval accuracy in our experiments is 80% with expectation maximized segmentations.

Keywords: image retrieval, geometric invariants, normalized histogram, hierarchical segmentation.

1 Introduction

Early versions of content-Based image retrieval (CBIR) systems [1,2] exploit global low-level features such as color, texture and shape as keys to retrieve images. However, the interest object may only occupy one small part of querying images, and the image retrieval approaches based global features can not achieve the expectation

* Work supported by Gansu Natural Science Foundation of China under Grant No. 3ZS051-A25-047.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 487–496, 2007.

results. Therefore, the schemes based local features of querying images were given in recent research works [3-4], i.e., the approaches based regions or objects of interest. So, the querying images should be firstly segmented into meaningful semantic objects. Although these works have been tried to decompose images into meaningful semantic objects, accurate object segmentation is still beyond current computer vision technique. Furthermore, the complexity of segmentation also brings bad feasibility to those approaches.

In order to conduct image retrieval with acceptable performance, we propose an effective retrieval scheme, which uses hierarchical multi-resolution segmentations (HMRS) as the substitutes of meaningful semantic segmentations. According to the hierarchical segmentations, the querying images are segmented into equal blocks with different shifts and sizes for different hierarchical levels. The smaller the block sizes, the higher the hierarchical segmentation levels. In this study, the proposed scheme is detailed and evaluated to conduct image retrieval with one novel region-based approach, i.e., the region-based scheme with wrapper segmentations [3], where the geometric invariants [5] and normalized histogram [6] are used as invariant features to different scales, orientations, and locations.

2 Hierarchical Multi-resolution Segmentations

The fundamental of hierarchical multi-resolution segmentations is aspired from the wavelet multi-resolution analysis, which can segment one image into equal size blocks to form an overlay of current images, while the image is overlapped by wavelet time-frequency atom in spatial frequency plane.

If one object in current images can fall into one block, the block size should be consistent with that of that object. However, objects in current images may present different scales, orientations, and locations. So, the image should also be segmented in different forms, and each segmentation form can be characterized by the shift and scale of blocks. Furthermore, the aspect ratios for different segmentation forms, is given as that of query images. With scale and rotation invariant features, objects can be retrieved by search the most similar blocks to queried images.

In order to formulate hierarchical multi-resolution segmentations, we denote s, r and c as the scale, the row and column shift of current segmentation form, which can described as one triple (s,r,c). The scale s is defined as the ratio of the segmentation sizes and the size of querying images. Then, the details of hierarchical multi-resolution segmentations can be shown by its flow map.

1. Compute the aspect ratio R of querying image.
2. Determine the maximal scale s_{max} of querying images. This can be drawn out by the maximal rectangle in current images with the same aspect ratio of the querying images.
3. Determine the minimal scale s_{min} of querying images that could be retrieved in current images. This is given by the object retrieval minimal scales in factual application requirements. In this paper, its value is given as the sixteenth of the original scale s_0 of querying images.
4. Determine the number L of the hierarchical levels and the segmentation shifts (r,c). Let Δs denote the ratio of the adjoining scales, which is kept constant for

hierarchical multi-resolution segmentation. By giving the discrete scales s between s_{max} and s_{min}, the number L is given by

$$L = \lfloor \log(s_{\min}/s_{\max}) / \log(\Delta s) \rfloor + 1 \tag{1}$$

where $\lfloor \cdot \rfloor$ denotes the largest integer of its operator. In this paper, Δs is given as 4/5. Furthermore, the segmentation shifts r and c are given the 4/5 height and length of current segmentation blocks, as the cropping 20% of querying images has almost the same invariant moment as the querying images [5].

5. According to the procedure, the hierarchical multi-resolution segmentation can be conducted at different levels and shifts.

At different segmentation levels, the margin parts of querying images will be discarded away after segmentations. Then, the scale and rotation invariant features are evaluated to conduct similarity matching for image retrieval.

3 Scale and Rotation Invariant Features

Objects in an image may present different scales and rotations from its query counterparts, so the required features have to be invariant to translation, scaling, and orientation. Many methods have been proposed to extract image invariants [7-9], such as zernike moments, rotation invariant Gabor and complex wavelet transform, rotation and scale invariant log-polar wavelet feature, discrete fourier transform moments, and so on. However, the order of zernike moments as image features, badly depends its image reconstruction accuracy, which will change with image contents. Other approaches involve log-polar transform, Gabor and wavelet transform, and invariant features can be exacted after complex image preprocessing. So, we adopt geometric invariants [5] and normalized histogram [6] as image features, which have simple implementation without image preprocessing.

3.1 Geometric Invariants

According to the results in literature[8], define the geometric moments $m(p,q)$ of a grayscale image $f(x,y)$ as

$$m(p,q) = \iint_{\Gamma} x^p y^q f(x,y)\, dxdy \tag{2}$$

where Γ is the support of the image. The central moments $\mu(p,q)$ are

$$\mu(p,q) = \iint_{\Gamma} (x-\bar{x})^p (y-\bar{y})^q f(x,y)\, dxdy \tag{3}$$

Here

$$\bar{x} = \frac{m(1,0)}{m(0,0)},\ \bar{y} = \frac{m(0,1)}{m(0,0)} \tag{4}$$

denote the centroid of an image. Define the normalized central moments $\eta(p,q)$

$$\eta(p,q)=\frac{\mu(p,q)}{\left[\mu(0,0)\right]^{\gamma}},\ \gamma=\frac{(p+q+2)}{2} \tag{5}$$

In the following, we present two sets of moment invariants. The first set is invariant to orthogonal transformations, while the second is invariant to general affine transformation. The following seven functions are invariant to orthogonal transformations

$$\begin{aligned}
\phi_1 &= \eta_{2,0}+\eta_{0,2}\\
\phi_2 &= \left(\eta_{2,0}-\eta_{0,2}\right)^2+4\eta_{1,1}^2\\
\phi_3 &= \left(\eta_{3,0}-3\eta_{1,2}\right)^2+\left(\eta_{0,3}-3\eta_{2,1}\right)^2\\
\phi_4 &= \left(\eta_{3,0}+\eta_{1,2}\right)^2+\left(\eta_{0,3}+\eta_{2,1}\right)^2\\
\phi_5 &= \left(\eta_{3,0}-3\eta_{1,2}\right)\left(\eta_{3,0}+\eta_{1,2}\right)\times\left[\left(\eta_{3,0}+\eta_{1,2}\right)^2-3\left(\eta_{2,1}+\eta_{0,3}\right)^2\right]\\
&\quad+\left(\eta_{0,3}-3\eta_{2,1}\right)\left(\eta_{0,3}+\eta_{2,1}\right)\times\left[\left(\eta_{0,3}+\eta_{2,1}\right)^2-3\left(\eta_{1,2}+\eta_{3,0}\right)^2\right]\\
\phi_6 &= \left(\eta_{2,0}-\eta_{0,2}\right)\left[\left(\eta_{3,0}+\eta_{1,2}\right)^2-\left(\eta_{0,3}+\eta_{2,1}\right)^2\right]\\
&\quad+4\eta_{1,1}\left(\eta_{3,0}+\eta_{1,2}\right)\left(\eta_{0,3}+\eta_{2,1}\right)\\
\phi_7 &= \left(3\eta_{2,1}-\eta_{0,3}\right)\left(\eta_{3,0}+\eta_{1,2}\right)\times\left[\left(\eta_{3,0}+\eta_{1,2}\right)^2-3\left(\eta_{2,1}+\eta_{0,3}\right)^2\right]\\
&\quad+\left(\eta_{3,0}-3\eta_{2,1}\right)\left(\eta_{2,1}+\eta_{0,3}\right)\times\left[\left(\eta_{0,3}+\eta_{2,1}\right)^2-3\left(\eta_{3,0}+\eta_{1,2}\right)^2\right]
\end{aligned} \tag{6}$$

It should be noted that ϕ_7 is invariant to reflection in the absolute value only. It is worth mentioning that scaling, rotation, and flipping are all considered within the class of orthogonal transformation. The following four functions are invariant under general affine transformation

$$\begin{aligned}
\varphi_1 &= \left[\mu_{2,0}\mu_{0,2}-\mu_{1,1}^2\right]/\mu_{0,0}^4\\
\varphi_2 &= \left[\mu_{3,0}^2\mu_{0,3}^2-6\mu_{3,0}\mu_{2,1}\mu_{1,2}\mu_{0,3}+4\mu_{3,0}\mu_{1,2}^3+4\mu_{2,1}^3\mu_{0,3}-3\mu_{2,1}^2\mu_{1,2}^2\right]/\mu_{0,0}^{10}\\
\varphi_3 &= \left[\mu_{2,0}\left(\mu_{1,2}\mu_{0,3}-\mu_{1,2}^2\right)-\mu_{1,1}\left(\mu_{3,0}\mu_{0,3}-\mu_{2,1}\mu_{1,2}\right)+\mu_{0,2}\left(\mu_{3,0}\mu_{1,2}-\mu_{2,1}^2\right)\right]/\mu_{0,0}^7\\
\varphi_4 &= [\mu_{2,0}^3\mu_{0,3}^2-6\mu_{2,0}^2\mu_{1,1}\mu_{0,3}-6\mu_{2,0}^2\mu_{0,2}\mu_{2,1}\mu_{0,3}+9\mu_{2,0}^2\mu_{0,2}\mu_{1,2}^2+12\mu_{2,0}\mu_{1,1}^2\mu_{2,1}\mu_{0,3}\\
&\quad+6\mu_{2,0}\mu_{1,1}\mu_{0,2}\mu_{3,0}\mu_{0,3}-18\mu_{2,0}\mu_{1,1}\mu_{0,2}\mu_{2,1}\mu_{1,2}-8\mu_{1,1}^3\mu_{3,0}\mu_{0,3}-6\mu_{2,0}\mu_{0,2}^2\mu_{3,0}\mu_{1,2}\\
&\quad+9\mu_{2,0}\mu_{0,2}^2\mu_{2,1}^2+12\mu_{1,1}^2\mu_{0,2}\mu_{3,0}\mu_{1,2}-6\mu_{1,1}\mu_{0,2}^2\mu_{3,0}\mu_{2,1}+\mu_{0,2}^3\mu_{3,0}^2]/\mu_{0,0}^{11}
\end{aligned} \tag{7}$$

Thus, the feature vector can be described as

$$\mathbf{M} = [\phi_1, \cdots, \phi_7, \varphi_1, \cdots, \varphi_4]^T \tag{8}$$

Where $(\cdot)^T$ represents matrix transpose.

Furthermore, if $\mathbf{M_0}$ denotes the feature vector of query image, the dissimilarity measure between query image and its queried image is given by Canberra distance metric between their feature vectors, which can achieve better image retrieval performance than other metrics [10].

$$D_m(\mathbf{M}_o, \mathbf{M}) = \sum_{i=1}^{11} \frac{|\mathbf{M}_o(i) - \mathbf{M}(i)|}{|\mathbf{M}_o(i)| + |\mathbf{M}(i)|} \tag{9}$$

where $\mathbf{M_0}$(i) and $\mathbf{M}$(i) denote the i-th element of $\mathbf{M_0}$ and $\mathbf{M}$, respectively.

According to this formula, the Canberra distance metric is normalized between 0 and 1. For color images, its moment invariants are implemented for red, green and blue components, and the correspondent dissimilarity measures are denoted as $D_m(r)$, $D_m(g)$, and $D_m(b)$. Then, their average is used as the dissimilarity measures of color images, that is

$$D_m = \frac{D_m(r) + D_m(g) + D_m(b)}{3} \tag{10}$$

3.2 Normalized Histogram

Suppose that $H_o(i)$ describes the normalized color histogram of query image with 256 bins, then the similar metric between histograms of query image and queried image, is given as the maximal absolute value of their cross-correlation function, i.e.,

$$D_h = \max_{m=-255,\cdots,0,\cdots,255} |R_{H_o,H}(m)| \tag{11}$$

where their cross-correlation functions $R_{Ho,H}(m)$ is defined as

$$R_{H_o,H}(m) = \begin{cases} \dfrac{1}{E(N-|m|)} \displaystyle\sum_{i=0}^{255} H_o(i+m) H(i) & m \geq 0 \\ R^*_{H_o,H}(-m) & m < 0 \end{cases} \tag{12}$$

Here E is a factor to normalize the cross-correlation functions at zero lag to 1, i.e.,

$$E = \sqrt{\sum_{i=0}^{255} [H_o(i)]^2} \cdot \sqrt{\sum_{i=0}^{255} [H(i)]^2} \tag{13}$$

As the cross-correlation function can cancel the effect of intensity on histogram, it is a more practical similar metric than histogram intersection [11]. Moreover, according to above formula, the cross-correlations functions at zero lag is normalized to the range [0,1], so the histogram similar metric is also normalized to the range [0,1].

3.3 Similarity Mmetric

If geometric invariants and normalized histogram are integrated, the similar metric of color images is normalized to [0,1] by the linear combination of that for geometric invariants and normalized histogram, i.e.,

$$D = \frac{D_m + (1 - D_h)}{2} \tag{14}$$

It can also be referred as the similar distance for color images when their geometric invariants and normalized histogram are synthetically considered. According to the similar metric, the images containing specific contents can be found by checking similar metric D whether smaller than a given similar metric threshold.

4 Image Retrieval Based-Hierarchical Segmentation

Image retrieval is generally conducted by searching the most similar images from databases to the query image, or region-based retrieval according to image segmentation results. The former has small complexity but bad retrieval accuracy, while the latter has good retrieval accuracy but great complexity. In order to balance computational complexity and retrieval accuracy, we use a flexible strategy for progressive image retrieval.

According to the HMRS procedure, an image is firstly segmented into blocks with different sizes at different levels. Then, the exaction of scale and rotation invariant features for all these blocks is followed. Once the similar metric of these blocks with the query objects are calculated, those blocks matching query objects will be retrieved with their scale and location information.

For given similarity metric threshold T_o, denote N(n) as the number of blocks in the n-th level segmentation. So, the L level HMRS-based image retrieval strategy can be formulated as following algorithm:

(1) Extract geometric invariants M_o and normalized histogram $H_o(i)$ of query image.
(2) Calculate similar metrics of different blocks
  For n = 0:L
    For m = 0: $N(n)$ -1
      (a)calculate the center coordinates $P(n,m)$ of sub-block $B(n,m)$.
      (b)extract geometric invariants $M(n,m)$ and normalized histogram $H(n,m)$.
      (c)calculate similarity metrics $s(n,m)$ between sub-blocks and query image
    End
  End
(3) Find those sub-blocks whose similarity metrics $s(n,m) \leq T_o$.
  If number of matching blocks ≥ 1
    Retrieve the image and mark the locations of these blocks.
    Retrieval succeeds
  Else
    Retrieval fails.
  End
(4) Image retrieval is over.

5 Experimental Results

To check the retrieval efficiency of proposed HMRS scheme, a test image database with 500 different size color images is constructed, which includes various types of images like flowers, birds, natural scenes, cars, etc. Fig.1 shows some image samples from the test databases with different sizes and categories.

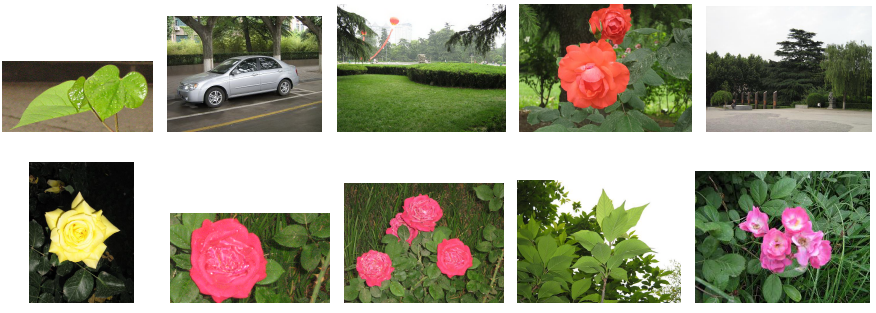


Fig. 1. Sample images from test image database

The retrieval performance is measured by two performance metrics, i.e., retrieval accuracy and retrieval rate. Retrieval accuracy is defined as the ratio of the number of retrieved relevant images to the total number of retrieved images, while the retrieval rate for the query image is measured by the ratio of the number of retrieved relevant images to the total number of query images in databases. They are effected by the similar metric thresholds and the levels of HMRS, i.e., unlike the retrieval accuracy, the retrieval rate will inversely change with similar metric threshold. Consequently, the optimal similar metric threshold can be determined by the balance of retrieval accuracy and retrieval rate through test experiments. When geometric invariants and normalized histogram are used as features to conduct the HMRS image retrieval scheme, their changes with similar metric threshold is displayed in Fig.2, and the threshold correspondent to their intersection is shown to be 0.175, which can balance retrieval accuracy and retrieval rate. The threshold is called the optimal similar metric threshold, whose correspondent retrieval accuracy and retrieval rate is evaluated to test the retrieval performance of HMRS image retrieval.

When we considered the first image in Fig.3 as a query image, the HMRS image retrieval scheme was conducted to find those images with the contents in the query image. Some most matching images were shown in Fig.3 as examples to test the retrieval results for the HMRS scheme, where the locations of the symbols, i.e., '+', 'o','*' and 'x', are the center coordinates of matching blocks in retrieved images for the 0, 1, 2, 3 level HMRS schemes, respectively. Thus, the information of the query about its sizes and locations in retrieved images is shown clearly, and this is also unique for the HMRS schemes when compared with other common image retrieval schemes.

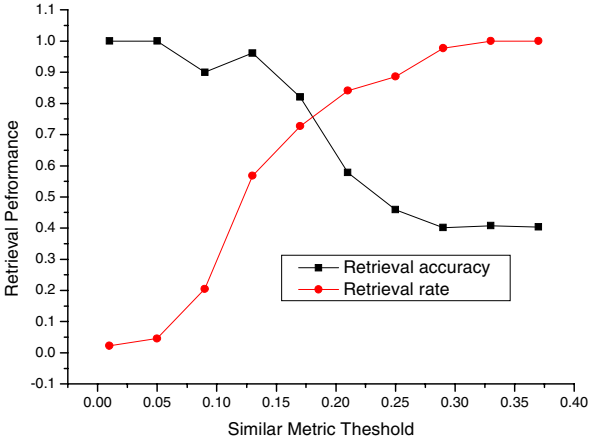


Fig. 2. The changes of retrieval accuracy and retrieval rate with similar metric threshold for 4 level HMRS image retrieval, when geometric invariants and normalized histograms are simultaneously used as features

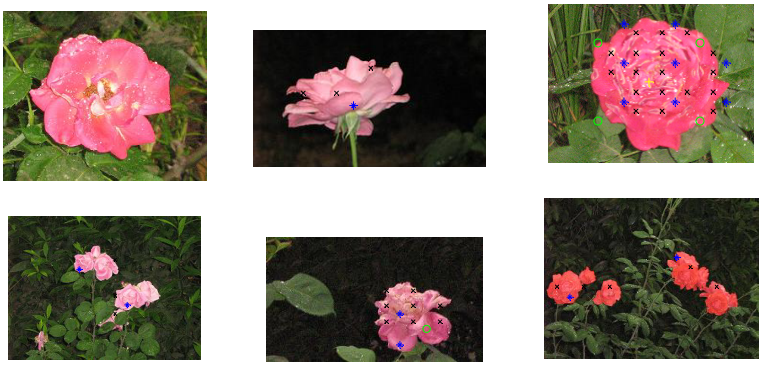


Fig. 3. Retrieval results of the first image via the HMRS image retrieval scheme with similar metric threshold 0.175, when geometric invariants and normalized histograms are used as features, while the sizes and locations of query images are also marked in retrieval results

Subsequently, we evaluated the retrieval performance of the HMRS scheme at different levels, where geometric invariants, normalized histograms and their combinations are used as features, respectively. In Fig.4, the retrieval accuracy in three different cases and at different levels is shown, when the optimal similar metric threshold is given. The zero HMRS image retrieval scheme is actually the common image retrieval scheme based on whole image matching. From Fig.4, it is clear that the retrieval accuracy with geometric invariants and normalized histograms as features is improved from 65% to 76%, while with geometric invariants from 40% to 78% and normalized histograms from 62% to 73%, respectively. Furthermore, at low levels, the retrieval performance of the scheme with geometric invariants is inferior to

that of the schemes with normalized histograms and their combinations, while the latter has approximate retrieval accuracy. However, with the increase of HMRS level number, its retrieval accuracy is improved quickly. As the balance of the normalized histogram scheme and the geometric invariant scheme, the HMRS image retrieval with their combinations at high levels can obtain better retrieval performance than that with normalized histograms but inferior to that with geometric invariants.

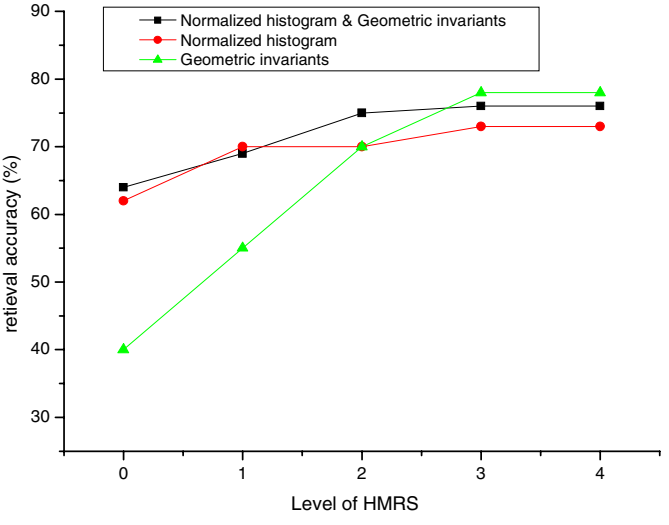


Fig. 4. Effect of the level number of HMRS image retrieval on retrieval performance of schemes with geometric invariants, normalized histograms and their combinations, respectively

Finally, the region-based image retrieval schemes [4] and the global texture-based schemes[7], are conducted to compare their performance with that of the proposed schemes in this paper. When these image retrieval schemes are implemented by Matlab 7.0 version at Intel Celeron 2.4 CPU platforms, it takes about 41.531 seconds, 0.219000 seconds and 27.750000 seconds respectively, to conduct one time image object search, where segmentation and feature exactions are also taken in account. Their average retrieval accuracy is given as 65%, 78% and 80% respectively, when about 10 time image retrieval accuracy were averaged. The complexity and inaccuracy of region-based image retrieval schemes, is the mainly limits for all current region-based retrieval schemes. Based the hierarchical multi-resolution segmentations, the proposed scheme given in this paper, can take good tradeoff between retrieval performance and system complexity.

6 Conclusion

In order to balance image retrieval accuracy and complexity, a feasible HMRS image retrieval scheme is proposed to conduct image retrieval with different size and rotation objects in images. It can be used a tradeoff between image retrieval based

whole matching and that regions-based schemes. The information about query objects in retrieved images with different sizes and locations can be returned via the scheme, which segments retrieved images into blocks with different sizes via a pyramid hierarchical multi-resolution segmentation process. The scheme was tested with geometric invariants, normalized histograms, and their combinations as image invariant features, respectively. As shown by experiments, the retrieval accuracy for common image retrieval can achieve comparable retrieval performance to that of region-based scheme in our experiments.

References

1. Tzagkarakis, G., Beferull-Lozano, B., Tsakalides, P.: Rotation-invariant texture retrieval with gaussianized steerable pyramids. IEEE Trans. Image Processing 15(9), 2702–2710 (2006)
2. Yap, P.T., Paramesran, R.: Content-based image retrieval using Legendre chromaticity distribution moments. IEE Proceedings of Vision, Image and Signal Processing 153(1), 24–77 (2006)
3. Farmer, M.E., Jain, A.K.: A wrapper-based approach to image segmentation and classification. IEEE Transactions on Image Processing 14(12), 2060–2072 (2005)
4. Pratikakis, I., Vanhamel, I., Sahli, H., Gatos, B., Perantonis, S.J.: Unsupervised watershed-driven region-based image retrieval. IEE Proceedings of Vision, Image and Signal Processing 153(3), 313–322 (2006)
5. Masoud, A., Ahmed, H.T.: Geometric Invariance in Image Watermarking. IEEE Trans. Image Processing 13(2), 145–153 (2004)
6. Kebin, J., Sheng, F., Qing, Z.: Rotation and translation invariant color image retrieval. In: Proceedings of 6th International Conference on Signal Processing, vol. 2, pp. 1063–1066 (August 2002)
7. Sim, D.-G., Kim, H.-K., Oh, D.-I.: Translation, scale, and rotation invariant texture descriptor for texture-based image retrieval. In: Proceedings of 2000 International Conference on Image Processing, vol. 3, pp. 742–745 (September 2000)
8. Vasconcelos, N.: On the complexity of probabilistic image retrieval. In: Proceedings of IEEE 8th International Conference on Computer Vision, vol. 2, pp. 400–407 (July 2001)
9. Shin, D., Kim, d., Kim, H., et al.: An image retrieval technique using rotationally invariant Gabor features and a localization method. In: Proceedings of 2003 International Conference on Multimedia and Expo, vol. 2, pp. 701–704 (July 2003)
10. Kokare, M., Chatterji, B.N., Biswas, P.K.: Comparison of similarity metrics for texture image retrieval. In: Proceedings of 2003 Asia-Pacific Region Conference on Convergent Technologies, vol. 2, pp. 571–575 (October 2003)
11. Kebin, J., Sheng, F., Qing, Z.: Rotation and translation invariant color image retrieval. In: Proceedings of 2002 6th International Conference on Signal Processing, vol. 2, pp. 1063–1066 (August 2002)

Knowledge Extraction from Unstructured Surface Meshes

Lars Graening, Markus Olhofer, and Bernhard Sendhoff

Honda Research Institute Europe GmbH,
Carl-Legien-Strasse 30, D-63073 Offenbach/Main, Germany
{lars.graening,markus.olhofer,bernhard.sendhoff}@honda-ri.de

Abstract. We propose methods that allow the investigation of local modifications of aerodynamic design data represented by discrete unstructured surface meshes. A displacement measure is suggested to evaluate local differences between the shapes. The displacement measure provides information on the amount and direction of surface modifications. Using the displacement measure in conjunction with statistical methods or data mining techniques provides meaningfull knowledge from the data set for guiding further shape optimization processes.

1 Introduction

In the field of 3D aerodynamic shape optimization, a large amount of geometric and flow field data is generated during the design process that usually encompasses several optimization runs, manual design phases and experiments. Typically, only the most promising results with regard to one or more possibly competing performance indices are exploited to define the overal result of the design process. However, a lot of information that could be condensed into comprehensive rules or observations concerning the design process in general is hidden in all of the data. Even poorly performing shapes can provide interesting insight into the fluid-dynamics of the problem and into the dynamics of the search process. This knowledge extracted from the large amount of data can be prepared in such a way that it can be used by the engineer or by an follow-up computational design and optimization processse. This type of knowledge extraction is the major focus of the present paper.

Obayashi et al. [10] were one of the first who addressed the problem of knowledge extraction from existing design data in order to gain some insights into the complex relationship between geometry and performance measurements. They used self organizing maps (SOM) in order to find groups of similar designs and for multicriteria performance improvements and tradeoffs. Although their methods have been applied to super sonic wing design, the data and design parameter sets were small, uniform and well defined. If different optimization runs have been performed with different design parameters, one first has to find an adequate representation which captures all shape variations and which can be applied to various data mining techniques.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 497–506, 2007.

Therefore, we suggest the use of unstructured surface meshes as a general representation for analyzing a given set of designs resulting from different shape optimization runs. Each optimization can be a manual or a computational process and can be based on different shape descriptors. The unstructured surface mesh as a general representation allows the analysis of local shape modifications and their influence on the performance value(s). In this paper, we will propose a displacement measure between surface meshes. The combination of the displacement measure with techniques from statistics and data mining allows the extraction of useful knowledge from the design database for the support of further optimization processes.

The paper is organized as follows. In Section 2, we will introduce unstructured surface meshes and outline the new displacement measure in Section 3, which captures local differences between designs. How we can extract knowledge from the displacement and performance data is described in Section 4. In the last section, we summarize the paper.

2 Surface Representation

For the optimization of the shape of three dimensional geometries often different parametric representations are used [2] which makes it difficult or even impossible to analyze the whole data set based on the applied parameterization. Therefore, we suggest unstructured triangular surface meshes as a general representation to describe the surface of each design. Most shape representations can be converted to unstructured surface meshes, see e.g. [3], [4].

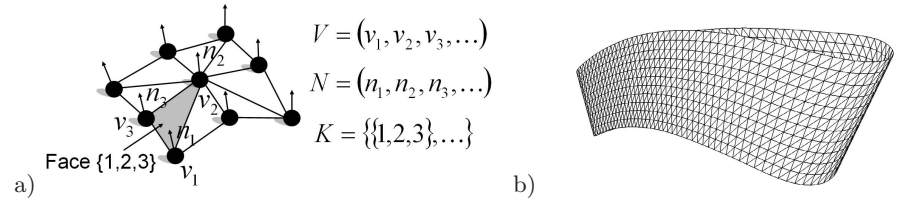


Fig. 1. Illustration of a) the specification of a triangular surface mesh $\mathcal{M} : (\mathcal{V}, \mathcal{N}, \mathcal{K})$ and b) an example of a triangular surface mesh for a 3D turbine blade

For the description of the surface mesh we start with the mathematical framework given in [5]. It is assumed that the shape of a 3D design is described using a polygonal surface mesh $\mathcal{M}$, which is a partially linear approximation of the contour of the design. We postulate that each mesh $\mathcal{M}$ consists of a list of vertices $\mathcal{V}$, a complex $\mathcal{K}$ and a list of normal vectors $\mathcal{N}$. The vertex list $\mathcal{V} = (\boldsymbol{v}_1, ..., \boldsymbol{v}_n)^T$ describes the geometric position of the vertices in $\mathbb{R}^3$, $\boldsymbol{v}_i = (x_1, x_2, x_3)^T$. A vertex can be seen as a sample point of the contour of the design. Each face of the polygonal surface mesh is defined by simplices of the form $\{i_1, i_2, i_3, ..., i_\mu\}$ where $i_l, l \in [1..n]$ are indices pointing to vertices that enclose the polygonal face.

Figure 1 a) illustrates a triangular surface mesh where the number of vertices, which are used to form each polygon, is set to 3. In addition to the vertex list, a list of normal vectors $\mathcal{N} = (\boldsymbol{n}_1, ..., \boldsymbol{n}_n)$ is given. Each normal vector $\boldsymbol{n}_i$ has a defined direction perpendicular to the surface mesh and provides local curvature information at the position of vertex $\boldsymbol{v}_i$. Figure 1 b) illustrates an example of a triangular surface mesh describing the contour of a 3D turbine blade design [1]. The normal vectors point towards the outside of the closed blade contour.

3 Displacement Measurement

Under the assumption that the surface triangulation results in surface meshes for which the location and the number of vertices is sufficiently precise to capture the characteristic changes of all designs in the given data set, the displacement is measured between each vertex on the reference design and each corresponding vertex on the modified design. In order to measure the displacement between two vertices of different surface meshes, the correspondence problem (which vertex from mesh $\mathcal{M}_r$ corresponds to which vertex from mesh $\mathcal{M}_m$) has to be solved and an appropriate metric has to be found for measuring the amount and the direction of the displacement between both vertices. We will not deal with the correspondence problem in this paper, the interested reader is refered to e.g. [7], [6]. In the following, we assume that two corresponding vertices have been identified.

3.1 Definition

The displacement measure should describe the position of a vertex in reference to another design. One way to capture this information is to use the difference vector $\boldsymbol{s}_{ij} = \boldsymbol{v}_i^r - \boldsymbol{v}_j^m$, which is the difference between vertex i of mesh $\mathcal{M}_r$ and vertex j of mesh $\mathcal{M}_m$. The difference vector clearly captures the correct displacement between both vertices. However, the difference vector is sensitive to possible errors resulting from wrong estimations of the corresponding points or from different sampling methods of the surfaces of the geometries. Furthermore, the difference vector requires $d = 3$ parameters for describing the displacement of one vertex in $\mathbb{R}^3$. Thus, to capture the displacement between two complete surface meshes the number of parameters is $3 \cdot n$, where n equals the number of vertices. To overcome the disadvantages of the difference vector, we suggest the following displacement measure:

$$\delta_{i,j}^{r,m} = \delta(\boldsymbol{v}_i^r, \boldsymbol{v}_j^m) = (\boldsymbol{v}_i^r - \boldsymbol{v}_j^m) \circ \boldsymbol{n}_i^r, \delta \in (-\infty, +\infty) \tag{1}$$

The displacement measure is defined as the projection of the difference vector $\boldsymbol{s}_{ij} = (\boldsymbol{v}_i^r - \boldsymbol{v}_j^m)$ onto the normal vector $\boldsymbol{n}_i$ of vertex $\boldsymbol{v}_i$ of the reference design $\mathcal{M}_r$. The absolute value of the displacement measure provides information on the amount of vertex modification while the sign of the displacement measure in conjuction with the normal vector of the vertex provides information on the direction of the vertex modification. The normal vector $\boldsymbol{n}_i$ points towards the normal or positive direction of vertex modification.

3.2 Major Properties

The displacement measure is by definition a vector quantity containing both the magnitude and the direction of vertex modification. If the modified vertex lies above the tangential plane described by the normal vector of the reference vertex, the displacement measure is positive, see Fig. 2 a). Whereas if the vertex lies below the tangential plane (Fig. 2 b) the displacement measure is negative. In the special case when the modified vertex is located within the tangential plane, the displacement measure is zero as shown in Fig. 2 c). If the reference vertex has been modified along the line described by the normal vector, the amount of the displacement measure equals the Euclidean distance between the reference and the modified vertex.

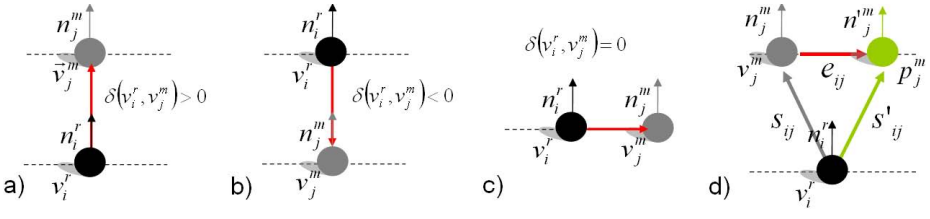


Fig. 2. Examples of the displacement measure. Figures a) and b) illustrate that a vertex displacement parallel (anti-parallel) to the normal direction results in a positive (negative) displacement value. A displacement perpendicular to the normal vector results in a displacement value of zero, as shown in c). Figure d) illustrates the error when calculating the displacement measure, which results from the discretization of the surface and the error when estimating corresponding points.

As Figure 2 d) indicates, the displacement value contains an error, which is mainly the result of the discretization of the surface using triangulation and of the correspondence problem. Formally, this can be written as

$$\delta_{i,j}^{r,m} = (\boldsymbol{s}_{ij} + \boldsymbol{e}_{ij}) \circ \boldsymbol{n}_i^r = \boldsymbol{s}_{ij} \circ \boldsymbol{n}_i^r + \boldsymbol{e}_{ij} \circ \boldsymbol{n}_i^r, \tag{2}$$

where $\boldsymbol{e}_{ij}$ describes the error between the ideal displacement value and the measured displacement value. Under the assumption that the curvature of both surfaces $\mathcal{M}_r$ and $\mathcal{M}_m$ is similar at the position of the corresponding vertices it follows that $\boldsymbol{n}_i^r \approx \boldsymbol{n}_j^m$. Then, the error term from equation 2 can be rewritten as:

$$\boldsymbol{e}_{ij} \circ \boldsymbol{n}_i^r \approx |\boldsymbol{e}_{ij}| cos(\angle(\boldsymbol{e}_{ij}, \boldsymbol{n}_j^m)). \tag{3}$$

If additionally a smooth surface or a small error $|\boldsymbol{e}_{ij}|$ is assumed, $\boldsymbol{e}_{ij}$ is perpendicular to $\boldsymbol{n}_j^m$ and hence $cos(\angle(\boldsymbol{e}_{ij}, \boldsymbol{n}_j^m)) \approx 0$. Thus the error term becomes zero. Therefore, the displacement measure is less sensitive to small errors arising from the surface triangulation or from an incorrect estimation of corresponding points.

Another advantage of the displacement measure compared with the difference vector is that only n parameters are required for the description of the differences between two unstructured surface meshes, where n equals the number of vertices.

4 Knowledge Extraction from Design Data

In aerodynamic design optimization the main goal is to find three dimensional shapes, which are optimal for specific performance measurements like aerodynamic drag or lift under specific constraints, e.g. manufacturing limitations. In general, during the optimization process a large number of shapes are generated and evaluated based on different representations and parameterizations. The result are heterogeneous design data sets from which only a very small number of designs are used in the end to determine the optimal shape (or a set of optimal shapes) which is processed further, e.g. in rapid prototyping devices for experiments. As we noted in the introduction, we aim at exploiting the information contained in the large remaining part of the data set. In this section, we describe how the displacement measure in conjunction with statistical and data mining methods can be used in order to extract meaningfull information (knowledge) from heterogeneous design data sets.

4.1 Displacement Analyzis

Analyzing local modifications in form of vertex displacement helps to gain some insight into the exploration of the design space. Two measures are suggested: the relative mean vertex displacement that provides information on how a vertex has been modified with respect to one reference design and the overall displacement variance that highlights the vertices which have been modified most frequently.

Relative Mean Vertex Displacement. In order to get information on local design modifications in reference to one baseline design, we define the *relative mean vertex displacement* :

$$\overline{\delta}_i^r = \frac{1}{N-1} \sum_{m=1, m \neq r}^{N} \delta_{i,j}^{r,m} \tag{4}$$

Given a data set of N unstructured surface meshes $\overline{\delta}_i^r$ evaluates the mean displacement of vertex j of all meshes m from the corresponding vertex i of the reference mesh (baseline) r. The measure provides information on how far a reference vertex has been modified along its normal vector with respect to the whole data set. If $\overline{\delta}_i^r > 0$, the vertex $\boldsymbol{v}_i^m$ has been modified parallel to the normal vector of the vertex and $\overline{\delta_i^r} < 0$ indicates a modification anti-parallel to the normal direction of the vertex. If $\overline{\delta}_i^r = 0$, the vertex has not been modified or the modifications around the reference vertex in the data set have canceled each other out. In order to identify the later situation one can calculate the variance of the deformation values. If there are outliers that affect the calculation of $\overline{\delta}_i^r$, we recommend to use the median instead of the mean in order to retrieve the desired information.

As an example the relative mean vertex displacement has been calculated based on a set of 200 turbine blades from different design optimization runs and a pre-selected references design. For illustration purpose the values have

been coded into corresponding color values and mapped onto the surface of the reference blade, Figure 3 a).

Overall Displacement Variance. In order to calculate $\overline{\delta}_i^r$, the baseline mesh r must be given. An alternative would be to calculate the mean displacement between all possible shape combinations in the data set. However, this is not sensible, because if the normal vectors of corresponding vertices are similar, it holds that $\delta_{i,j}^{r,m} \approx -\delta_{i,j}^{m,r}$ and as a result such a measure would always tend to zero.

In order to get an overview over the variations of local design modifications an *overall displacement variance* can be defined as follows:

$$\sigma_{\delta_i} = \sqrt{\sum_{r=1}^{N} \sum_{m=1,m\neq r}^{N} (\delta_{i,j}^{r,m} - \overline{\delta}_i)^2} \approx \sqrt{2 \cdot \sum_{l=1}^{N} \sum_{m=r+1}^{N} (\delta_{i,j}^{r,m})^2} \tag{5}$$

This measure describes the strength and the frequency of local design modifications based on the whole data set. Following our argument above, we can set $\overline{\delta}_i \approx 0$.

Figure 3 b) shows the overall displacement for the turbine blade data set. In order to visualize high as well as low variances σ_{δ_i} has been displayed in logarithmic scale.

4.2 Sensitivity Analysis

Sensitivity analysis relates the displacement measure to variations of the corresponding performance values.

Relative Vertex Correlation Coefficient. The *relative vertex correlation coefficient* R_i^r, see Equation 6, formalizes the linear relation between local modifications in form of vertex displacements and performance values with respect to a chosen reference design. $\phi^{r,m} = f^r - f^m$ is the performance difference between two designs r and m and $\overline{\phi}^r$ is the mean value of the performance differences with respect to the reference design r.

$$R_i^r = \frac{\sum_{m=1,m\neq r}^{N} (\delta_{i,j}^{r,m} - \overline{\delta}_i^r)(\phi^{r,m} - \overline{\phi}^r)}{\sigma_{\delta_i^r} \sigma_{\phi^r}} \tag{6}$$

$$\sigma_{\delta_i^r} = \sqrt{\sum_{m=1,m\neq r}^{N} (\delta_{i,j}^{r,m} - \overline{\delta}_i^r)^2}, \qquad \overline{\phi}^r = \frac{1}{N} \sum_{m=1,m\neq r}^{N} \phi^{r,m} \tag{7}$$

$R_i^r > 0$ indicates that moving the vertex parallel to the normal vector is most likely to improve the performance of the design and vice versa. Again two situations can lead to a vanishing R_i^r value. Firstly, the obvious explanation is that an (anti)-parallel modification of the vertex has no effect on the performance measure. Secondly, if the vertex is already located in an optimal position,

every modification will reduce the performance and R_i^r will also be close to zero. In order to distinguish between both cases, one could fit a linear model to the displacement and performance difference pairs and calculate the residual of the linear model. This residual provides information on the uncertainty of the correlation coefficient. Of course, the uncertainty of the correlation coefficient might also result from noisy data or non-linear relations between displacement measure and performance differences.

Concerning the 3D turbine blade example the performance of each design is determined by the overall pressure loss of the blade, see [1]. Based on the displacement and pressure loss data the relative vertex correlation has been calculated. The result is illustrated in Figure 3.

Vertex Sensitivity. In order to identify vertices that are sensitive to performance changes based on the whole data set without referring to one baseline shape, the Pearson correlation coefficient is calculated based on all pairwise design comparisons. Calculating the mean value for all performance differences obviously results in $\overline{\phi} = 0$. We define the *overall vertex correlation coefficient* as follows (assuming again $\overline{\delta}_i \approx 0$):

$$R_i = \frac{\sum_{r=1}^{N} \sum_{m=1, m \neq r}^{N} \delta_{i,j}^{r,m} \phi^{r,m}}{\sigma_{\delta_i} \sigma_\phi}, \qquad \sigma_\phi = \sqrt{2 \cdot \sum_{r=1}^{N} \sum_{m=r+1}^{N} (\phi^{r,m})^2} \tag{8}$$

The overall vertex correlation coefficient captures the linear relationship between the displacement and performance changes. In order to be less sensitive to outliers or noise in the data, it is reasonable to apply the Spearman rank based coefficient instead of the Pearson correlation coefficient. Since the overall vertex correlation is linear, information is provided to distinguish between those vertices which are more likely to improve the performance by moving them parallel to the direction of the normal vector and those which improve the performance when moving them anti-parallel to the direction of the normal vector.

The overall vertex correlation coefficient has also been calculated for the turbine blade data set. In order to identify most sensitive points a threshold has been applied to the sensitivity values, R_i. The emerged regions can be distinguished into regions of positive and those of negative correlation as shown in Figure 3 d).

In aerodynamic design optimization the interrelation between design modifications and performance changes is often highly non-linear. In order to capture also non-linearities, one can apply information based measures like *mutual information* [11] to determine the sensitivity of vertices. The disadvantage of non-linear methods like mutual information is that the information to predict the direction of design improvement with respect to the normal vector is lost.

4.3 Modeling and Analyzing Interrelations

For the calculation of the measures described above, the displacement of each vertex is considered independent of the others. Especially in aerodynamics the

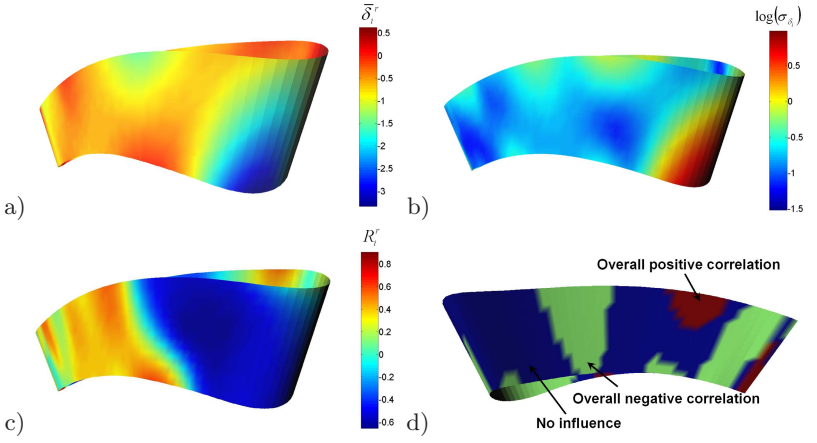


Fig. 3. As an example the suggested methods are calculated based on a data set which consists of 200 3D turbine blades: a) Relative mean vertex displacement, b) Overall displacement variance in logarithmic scale, c) Relative vertex correlation coefficient, d) Sensitive regions of the blade

interrelation between distant vertices or design regions and their joint influence on the performance plays an important role. In this section, special characteristics for the extraction of knowledge in form of associative rules based on data from unstructured surface meshes are discussed and illustrated by means of the blade example described above. The rules describe the relation between the displacement of distant vertices and their joint influence on the performance criteria. Modeling the interrelation between input variables is achieved by applying well known modeling techniques like Fuzzy rule induction, Bayesian networks, decision trees or others to the data set, for an overview of techniques see e.g. [8].

Rule Induction. Generally, the number of input parameters must be kept small for most modeling techniques in order to produce a small set of interpretable and manageable association rules. With respect to the used shape representation, the number of inputs equals the number of vertices n, which is large in practice. Therefore, a reduction of the number of input parameters is strongly required. Concerning the present turbine blade example, this process consists of the following steps:

1. Perform sensitivity analysis
2. Select most sensitive vertices (e.g. apply threshold to the sensitivity metrics)
3. Cluster sensitive vertices to form sensitive areas (e.g. K-means)
4. Calculate cluster centers of the sensitive areas (the resulting number of input variables strongly depends on the number of clusters which are defined for the clustering algorithm)
5. Use displacement of vertices closest to cluster centers for rule extraction

The main task when modeling interrelations between distant design regions or vertices is to extract associative rules which can be interpreted by aerodynamic

engineers. These rules are subdivided into relative rules, which refer to a baseline shape, and general rules, which refer to the complete data set.

Besides standard real-valued input for the modeling technique, the input can also be restricted to the sign of the displacement measure. In this case, rules from the two-valued input describe the interrelation between the direction of vertex displacement and the change in the performance value.

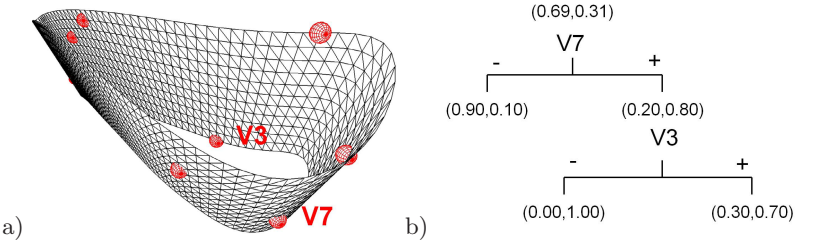


Fig. 4. a) Illustration of the vertices close to the cluster centers of the sensitive areas. b) Simplified decission tree describing the joint interrelation between a subset of vertices and the influence to the preformance (overall pressure loss).

Figure 4 illustrates the reduced subset of parameters (vertices) and a part of the complete classification tree extracted from the turbine blade data set. The classification tree describes the interrelation between the direction of vertex displacement and the change in the overall pressure loss. The rules for the correlated movement of vertices are extracted in form of joint probabilities. For example, moving vertex V7 alone will improve the performance by an probability of $p(\phi > 0|\delta^r_{V7} > 0) = 0.80$. But moving V7 correlated with V3 increases the probability to $p(\phi > 0|\delta^r_{V7} > 0, \quad \delta^r_{V3} < 0) = 1.00$ for improving the overall pressure loss.

5 Summary

In this paper, we investigated the problem of how to extract knowledge from the large heterogeneous data sets that usually result from aerodynamic shape optimization processes. Firstly, the aim is to communicate this knowledge to the engineer to increase his/her understanding of the relation between shape and aerodynamic performance, e.g. which part of the design space has been explored and which part has been largely ignored in the past design processes. Secondly, the information from the data set can be used in order to improve the ongoing optimization process, e.g. by specifying parameters of the optimization algorithms or by increasing the generalization capabilities and reducing the approximation erros of surrogate models [13].

The main contribution described in this paper is the formulation of a displacement measure that acts on a generalized shape representation - the unstructured mesh. Based on the displacement measure a number of methods and approaches

for displacement analysis, sensitivity analysis and rule extraction were suggested and formulized.

In order to demonstrate the feasibility of the suggested approach, we have shown examples for the proposed measures from a data set taken from the optimization of a 3D turbine blade.

References

1. Hasenjäger, M., Sendhoff, B., Sonoda, T., Arima, T.: Three dimensional aerodynamic optimization for an ultra-low aspect ratio transonic turbine stator blade. In: Proceedings of the ASME Turbo Expo (2005)
2. Samareh, J.: A Survey of Shape Parametrization Techniques. In: Intern. Forum on Aeroelasticity and Structural Dynamics Conf. (1999)
3. Ito, Y., Nakahashi, K.: Surface Triangulation for Polygonal Models Based on CAD Data. Intern. Journal for Numerical Methods in Fluids 39 (2002)
4. Shimada, K.: Anisotropic Triangular Meshing of Parametric Surfaces via Close Packing of Ellipsoidal Bubbles. Intern. Journal of Computational Geometries and Applications 10(4), 400–424 (2000)
5. Alexa, M.: Recent Advances in Mesh Morphing. Computer Graphics forum 21(2), 173–197 (2002)
6. Jain, V., Zhang, H.: Robust 3D Shape Correspondence in the Spectral Domain, IEEE Intern. In: SMI 2006. IEEE Intern. Conf. on Shape Modeling and Applications, p. 19 (2006)
7. Wang, Y., Peterson, B.S., Staib, L.H.: Shape-Based 3D Surface Correspondence Using Geodesics and Local Geometry. Computer Vision and Pattern Recognition, 644–651 (2000)
8. Maimon, O., Rokach, L. (eds.): The Data Mining and Knowledge Discovery Handbook. Springer, Heidelberg (2005)
9. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth, Inc., CA (1984)
10. Obayashi, S., Sasaki, D.: Visualization and Data Mining of Pareto Solutions Using Self-Organizing Map. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, p. 71. Springer, Heidelberg (2003)
11. Steuer, R., Kurths, J., Daub, C.O., Weise, J., Selbig, J.: The Mutual Information: Detecting and Evaluating Dependencies between Variables. Bioinformatics (2002)
12. Cohen, J.: Statistical Power Analysis for the Behavioral Sciences, 2nd edn. Lawrence Erlbaum Associates, Hillsdale, NJ (1988)
13. Jin, Y.: A Comprehensive Survey of Fitness Approximation in Evolutionary Computation. Soft Computing 9(1), 3–12 (2005)

Clustering with Reinforcement Learning

Wesam Barbakh and Colin Fyfe

The University of Paisley,
Scotland
wesam.barbakh, colin.fyfe@paisley.ac.uk

Abstract. We show how a previously derived method of using reinforcement learning for supervised clustering of a data set can lead to a sub-optimal solution if the cluster prototypes are initialised to poor positions. We then develop three novel reward functions which show great promise in overcoming poor initialization. We illustrate the results on several data sets. We then use the clustering methods with an underlying latent space which enables us to create topology preserving mappings. We illustrate this method on both real and artificial data sets.

1 Introduction

We have previously investigated [2, 4] clustering methods which are robust with respect to poor initialization. In this paper, we investigate using reinforcement learning with reward functions which are related to our previous clustering methods in that the reward functions also allow the reinforcement learning algorithms to overcome the disadvantages of a poor initialization and achieve the globally optimal clustering.

We begin by reviewing a reinforcement learning algorithm which has previously [9] been used to perform clustering of data sets.

2 The Bernoulli Model

[12, 11] investigated a particular form of reinforcement learning in which reward for an action is immediate which is somewhat different from mainstream reinforcement learning [10, 7]. Williams [11] considered a stochastic learning unit in which the probability of any specific output was a parameterised function of its input, $\mathbf{x}$. For the i^{th} unit, this gives

$$P(y_i = \zeta | \mathbf{w}_i, \mathbf{x}) = f(\mathbf{w}_i, \mathbf{x}) \tag{1}$$

where, for example,

$$f(\mathbf{w}_i, \mathbf{x}) = \frac{1}{1 + \exp(- \parallel \mathbf{w}_i - \mathbf{x} \parallel^2)} \tag{2}$$

Williams [11] considers the learning rule

$$\Delta w_{ij} = \alpha_{ij}(r_{i,\zeta} - b_{ij}) \frac{\partial \ln P(y_i = \zeta | \mathbf{w}_i, \mathbf{x})}{\partial w_{ij}} \tag{3}$$

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 507–516, 2007.

where α_{ij} is the learning rate, $r_{i,\zeta}$ is the reward for the unit outputting ζ and b_{ij} is a reinforcement baseline which in the following we will take as the reinforcement comparison, $b_{ij} = \overline{r} = \frac{1}{K}\sum r_{i,\zeta}$ where K is the number of times this unit has output ζ. ([11], Theorem 1) shows that the above learning rule causes weight changes which maximises the expected reward.

[11] gave the example of a Bernoulli unit in which $P(y_i = 1) = p_i$ and so $P(y_i = 0) = 1 - p_i$. Therefore

$$\frac{\partial \ln P(y_i)}{\partial p_i} = \begin{cases} -\frac{1}{1-p_i} & \text{if } y_i = 0 \\ \frac{1}{p_i} & \text{if } y_i = 1 \end{cases} = \frac{y_i - p_i}{p_i(1-p_i)} \tag{4}$$

[9] applies the Bernoulli model to (unsupervised) clustering with

$$p_i = 2(1 - f(\mathbf{w}_i, \mathbf{x})) = 2\left(1 - \frac{1}{1 + \exp(- \parallel \mathbf{w}_i - \mathbf{x} \parallel^2)}\right) \tag{5}$$

The environment identifies the p_{k*} which is maximum over all output units and y_{k*} is then drawn from this distribution. Rewards are given such that

$$r_i = \begin{cases} 1 & \text{if } i = k* \text{ and } y_i = 1 \\ -1 & \text{if } i = k* \text{ and } y_i = 0 \\ 0 & \text{if } i \neq k* \end{cases} \tag{6}$$

where $k*$ represents the winning node, the node that are most similar to the input sample. This is used in the update rule

$$\Delta w_{ij} = \alpha r_i (y_i - p_i)(x_j - w_{ij}) \tag{7}$$

$$= \alpha |y_i - p_i| (x_j - w_{ij}) \text{ for } i = k* \tag{8}$$

which is shown to perform clustering of the data set.

Implementation

1. Randomly select a sample $\mathbf{x}$ from the data set.
2. For $i = 1, ..., L$ compute the probability p_i
3. Specify the winning unit $k*$ with $p_{k*} = max p_i$, and sample the output y_{k*} from p_{k*}
4. Compute the reinforcement rewards r_{k*} using equation (6)
5. Update the weight vectors w_{k*} using equation (7)
6. Repeat until convergence

2.1 Simulation

We applied the Bernoulli algorithm to the artificial data set shown in Figure 1, left, but the Bernoulli algorithm failed to identify all the clusters successfully as shown in Figure 1, right.

The Bernoulli algorithm is sensitive to the prototypes' initialization which can lead it to finding local optima which often are detectable because of dead prototypes which are not near any data. The main reason for these problems is that we update the winner prototypes only, not all of them.

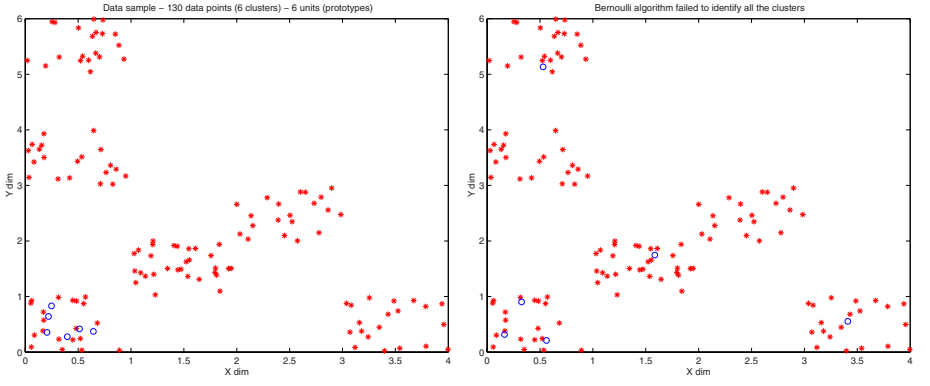


Fig. 1. Left: artificial data set is shown as 6 clusters of red '*'s, and 6 prototypes of blue 'o's. Right: Bernoulli algorithm failed to identify all the clusters successfully.

3 Algorithm RL1

A first new algorithm, RL1, has the following reward function:

$$r_i = \begin{cases} \frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^3}{\|\mathbf{x}-\mathbf{m}_i\|^3} & \text{if } y_i = 1 \\ -\frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^3}{\|\mathbf{x}-\mathbf{m}_i\|^3} & \text{if } y_i = 0 \end{cases} \forall i \tag{9}$$

where

$$k* = \arg\min_{k=1}^{K}(\| \mathbf{x} - \mathbf{m}_k \|)$$

This new reward function has the following features:

1. We apply this equation to all prototypes, not only the winners and thus all prototypes can find the clusters even if they are initialized badly.
2. This reward function allows the prototypes to respond differently to each other, and each prototype before moving to any new location responds to all the other prototypes' position, and hence it is possible for it to identify the free clusters that are not recognized by the other prototypes.
3. This reward function gives the highest value, 1, for highest similarity between the data point and the node (prototype).

3.1 Simulation

Figure 2 shows the result after applying RL1 algorithm to the artificial data set, but with very poor prototypes' initialization.

Figure 2, left shows the prototypes after many iterations but before convergence; in this Figure we can see one prototype still far from the data points while others have spread into data; this distant prototype still has the ability to learn even if it is very far from data, and this is an advantage for this algorithm over the previous algorithms.

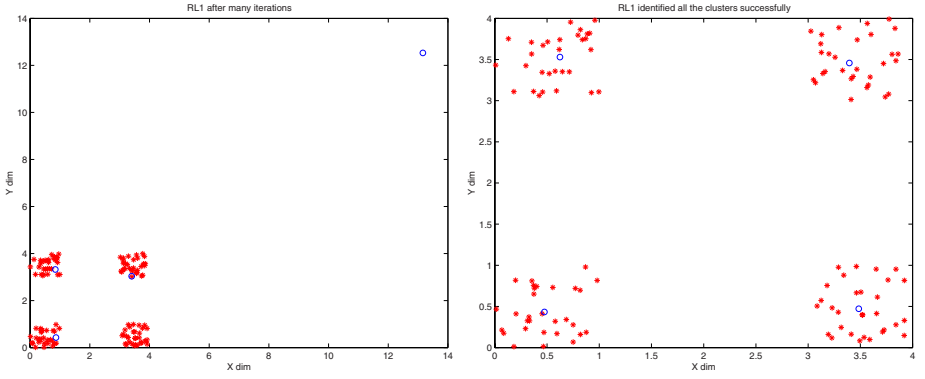


Fig. 2. Left: RL1 result after many iterations but before convergence. Right: RL1 result after convergence.

4 New Algorithm RL2

A second new algorithm, RL2, has the following reward function:

$$r_i = \begin{cases} 1 & \text{if } i = k^* \text{ and } y_i = 1 \\ \frac{1-\exp(-\beta\|\mathbf{x}-\mathbf{m}_{k*}\|^3)}{\|\mathbf{x}-\mathbf{m}_i\|^3} & \text{if } i \neq k^* \text{ and } y_i = 1 \\ -1 & \text{if } i = k^* \text{ and } y_i = 0 \\ \frac{\exp(-\beta\|\mathbf{x}-\mathbf{m}_{k*}\|^3)-1}{\|\mathbf{x}-\mathbf{m}_i\|^3} & \text{if } i \neq k^* \text{ and } y_i = 0 \end{cases} \tag{10}$$

where again $k* = \arg\min_j \parallel \mathbf{x} - \mathbf{m}_j \parallel$.

The reward function (10) has values ranged between 0 and 1. We update the closest prototype (or most similar one) by giving it directly a maximum possible reward value, 1, to allow it to learn more than others and also to a void any division by zero which may happen using the second equation in (10). The second equation in (10) is used for all the other prototypes. Prototypes closer (or more similar) to the input data sample will learn more than others by taking higher reward value, and so on for all prototypes.

5 Algorithm RL3

A third new algorithm, RL3, has the following reward function:

$$r_i = \begin{cases} \frac{1}{\|\mathbf{x}-\mathbf{m}_i\|^4\{\sum_{l=1}^{K}\frac{1}{\|\mathbf{x}-\mathbf{m}_l\|^2}\}^2} & \text{if } y_i = 1 \\ \frac{-1}{\|\mathbf{x}-\mathbf{m}_i\|^4\{\sum_{l=1}^{K}\frac{1}{\|\mathbf{x}-\mathbf{m}_l\|^2}\}^2} & \text{if } y_i = 0 \end{cases} \tag{11}$$

The reward function in (11) has similar principles like the previous new reward functions. It has values ranged between 0 and 1. All the prototypes can learn in an effective way. The prototype that is more similar to the input data sample takes higher reward value. In implementation, to avoid any division by zero we can rewrite (11) as follows:

$$r_i = \begin{cases} \dfrac{\frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^4}{\|\mathbf{x}-\mathbf{m}_i\|^4}}{\left\{1+\sum_{l\neq k*}^{K}\frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^2}{\|\mathbf{x}-\mathbf{m}_l\|^2}\right\}^2} \text{ if } y_i = 1 \\ \dfrac{-\frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^4}{\|\mathbf{x}-\mathbf{m}_i\|^4}}{\left\{1+\sum_{l\neq k*}^{K}\frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^2}{\|\mathbf{x}-\mathbf{m}_l\|^2}\right\}^2} \text{ if } y_i = 0 \end{cases} \tag{12}$$

where $k* = \arg\min_j \| \mathbf{x} - \mathbf{m}_j \|$. Note that $\frac{\|\mathbf{x}-\mathbf{m}_{k*}\|^4}{\|\mathbf{x}-\mathbf{m}_{k*}\|^4}$ is always set to 1.

5.1 Simulation

Figure 3 shows the results after applying Bernoulli algorithm, top right, RL2, bottom left, and RL3, bottom right, to the artificial data set shown in Figure 3, top left. RL2 and RL3 succeeded to identify the clusters successfully while Bernoulli model failed.

6 A Topology Preserving Mapping

In this section, we show how we can extend RL1 and RL2 to provide new algorithms for visualisation and topology-preserving mappings.

6.1 RL1 Topology-Preserving Mapping (RL1ToM)

A topographic mapping (or topology preserving mapping) is a transformation which captures some structure in the data so that points which are mapped close to one another share some common feature while points which are mapped far from one another do not share this feature. The Self-organizing Map (SOM) was introduced as a data quantisation method but has found at least as much use as a visualisation tool.

Topology-preserving mappings such as the Self-organizing Map (SOM) [8] and the Generative Topographic Mapping(GTM) [5] have been very popular for data visualization: we project the data onto the map which is usually two dimensional and look for structure in the projected map by eye. We have recently investigated a family of topology preserving mappings [6] which are based on the same underlying structure as the GTM.

The basis of our model is K latent points, $t_1, t_2, \cdots, t_K$, which are going to generate the K prototypes, $\mathbf{m}_k$. To allow local and non-linear modeling, we map those latent points through a set of M basis functions, $f_1(), f_2(), \cdots, f_M()$. This gives us a matrix Φ where $\phi_{kj} = f_j(t_k)$. Thus each row of Φ is the response of

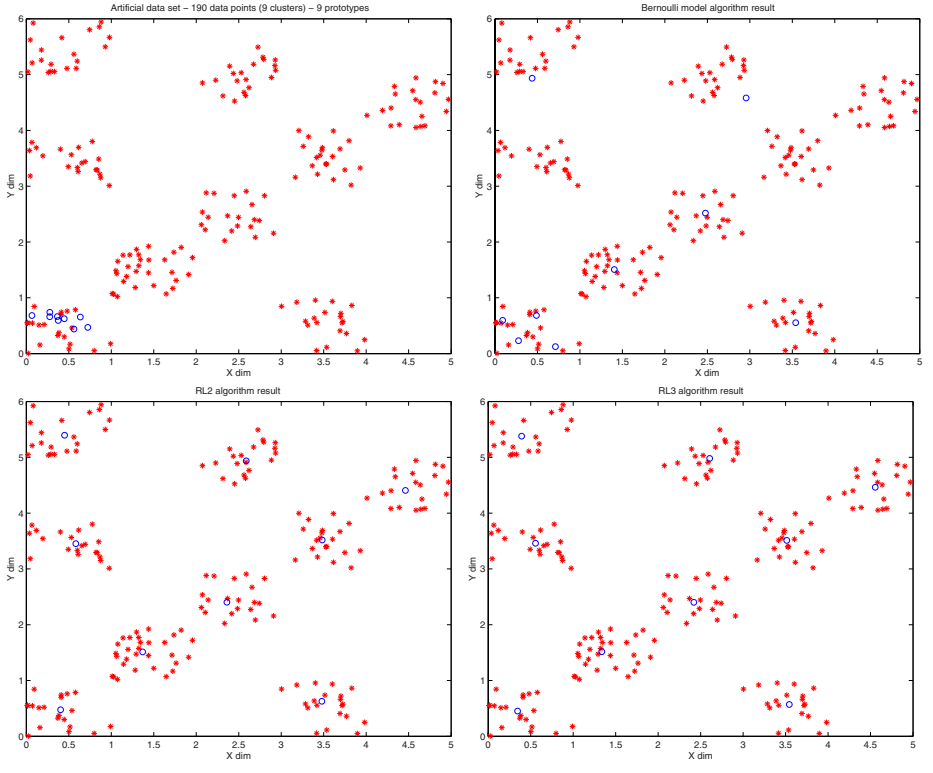


Fig. 3. Top left: Artificial data set with poor prototypes' initialization. Top right: Bernoulli algorithm result. Bottom left: RL2 algorithm result. Bottom right: RL3 algorithm result.

the basis functions to one latent point, or alternatively we may state that each column of Φ is the response of one of the basis functions to the set of latent points. One of the functions, $f_j()$, acts as a bias term and is set to one for every input. Typically the others are gaussians centered in the latent space. The output of these functions are then mapped by a set of weights, W, into data space. W is $M \times D$, where D is the dimensionality of the data space, and is the sole parameter which we change during training. We will use $\mathbf{w}_i$ to represent the i^{th} column of W and Φ_j to represent the row vector of the mapping of the j^{th} latent point. Thus each basis point is mapped to a point in data space, $\mathbf{m}_j = (\Phi_j W)^T$.

We may update W either in batch mode or with online learning: with the Topographic Product of Experts [6], we used a weighted mean squared error; with the Inverse Exponential Topology Preserving Mapping [1], we used Inverse Exponential K-means, with the Inverse-weighted K-means Topology-preserving Mapping (IKToM) [3, 2], we used Inverse Weighted K-means (IWK). We now apply the RL1 algorithm to the same underlying structure to create a new topology preserving algorithm.

Each data point is visualized as residing at the prototype on the map which would win the competition for that data point. However we can do rather better by defining the responsibility that the j^{th} prototype has for the i^{th} data point as

$$r_{ji} = \frac{\exp(-\gamma \parallel \mathbf{x}_i - \mathbf{w}_j \parallel^2)}{\sum_k \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{w}_k \parallel^2)} \tag{13}$$

We then project points taking into account these responsibilities: let y_{ij} be the projection of the i^{th} data point onto the j^{th} dimension of the latent space; then

$$y_{ij} = \sum_k t_{kj} r_{ki} \tag{14}$$

where t_{kj} is the j^{th} coordinate of the k^{th} latent point.

6.2 RL2 Topology-Preserving Mapping (RL2ToM)

RL2ToM algorithm like RL1ToM has the same structure as the GTM, with a number of latent points that are mapped to a feature space by M Gaussian functions, and then into the data space by a matrix W. Each latent point t indexed by k is mapped, through a set of M fixed basis functions $\phi_1()$, $\phi_2()$,...,$\phi_M()$ to a prototype in data space $m_k = W\phi(t_k)$. But the similarity ends there because the objective function is not a probabilistic function like the GTM neither it is optimised with the Expectation-Maximization (EM) algorithm. Instead, the RL2ToM uses the well proved clustering abilities of the K-means algorithm, improved by using RL2 to make it insensitive to initialisation.

6.3 Simulation

6.3.1 Artificial Data Set

We create a simulation with 20 latent points deemed to be equally spaced in a one dimensional latent space, passed through 5 Gaussian basis functions and

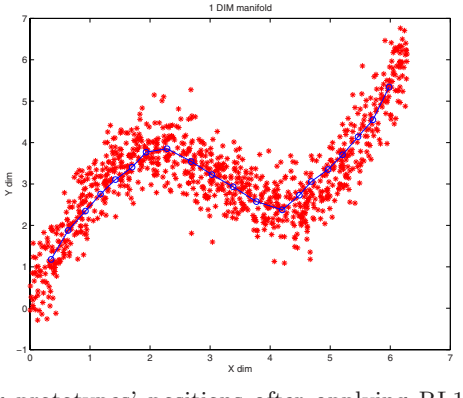


Fig. 4. The resulting prototypes' positions after applying RL1ToM. Prototypes are shown as blue 'o's.

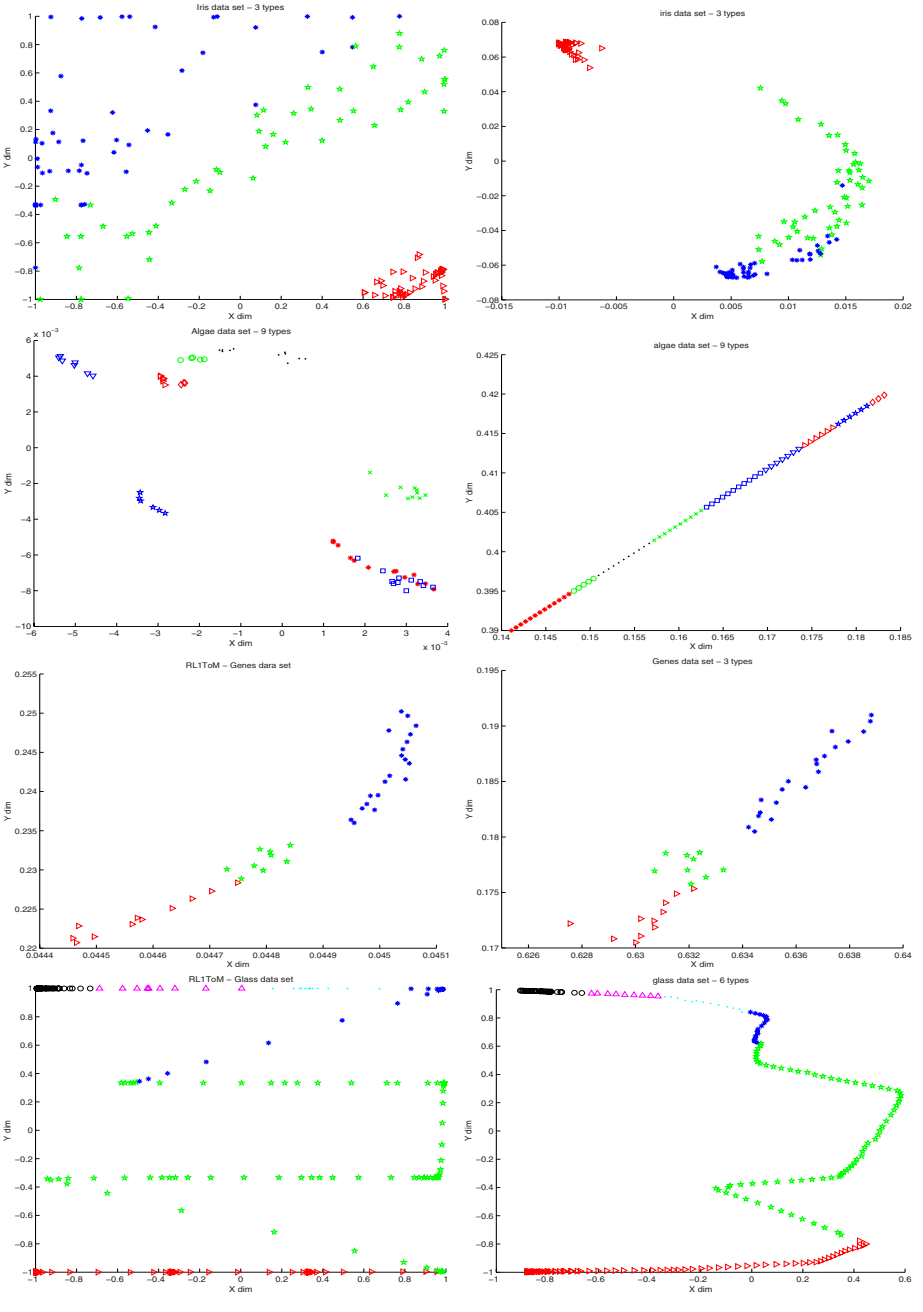


Fig. 5. The first column shows the results of using RL1ToM; the second column shows the results of using RL2ToM. The top line is the projection of the iris data set; the second line shows the algae data set; the third line shows the genes data set; the bottom line shows the glass data set.

then mapped to the data space by the linear mapping W which is the only parameter we adjust. We generated 500 two dimensional data points, (x_1, x_2), from the function $x_2 = x_1 + 1.25\sin(x_1) + \mu$ where μ is noise from a uniform distribution in [0,1]. Final result from the RL1ToM is shown in Figure 4.

6.3.2 Real Data Set

Iris data set: 150 samples with 4 dimensions and 3 types.
Algae data set: 72 samples with 18 dimensions and 9 types
Genes data set: 40 samples with 3036 dimensions and 3 types
Glass data set: 214 samples with 10 dimensions and 6 types

We show in Figure 5, left and right, the projections of the real data sets onto a two dimensional grid of latent points using RL1ToM and RL2ToM, respectively. The results are comparable with others we have with these data sets from a variety of different algorithms.

7 Conclusion

We have shown how reinforcement learning of cluster prototypes can be performed robustly by altering the reward function associated with finding the clusters. We have illustrated three different reward functions which clearly have a family resemblance. Most importantly all three overcome the disadvantages of poor initialization in that they do not succumb to local minima as the existing Bernoulli algorithm does.

We have also illustrated how a topology preserving mapping can be created by using these algorithms with an underlying fixed latent space. Future work will compare these methods with our existing methods of creating robust optimal clusters.

References

[1] Barbakh, W.: The family of inverse exponential k-means algorithms. Computing and Information Systems 11(1), 1–10 (2007)
[2] Barbakh, W., Crowe, M., Fyfe, C.: A family of novel clustering algorithms. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 283–290. Springer, Heidelberg (2006)
[3] Barbakh, W., Fyfe, C.: Performance functions and clustering algorithms. Computing and Information Systems 10(2), 2–8 (2006)
[4] Barbakh, W., Fyfe, C.: Tailoring local and global interactions in clustering algorithms. Technical Report 40, School of Computing, University of Paisley (March 2007), ISSN 1461-6122
[5] Bishop, C.M., Svensen, M., Williams, C.K.I.: Gtm: The generative topographic mapping. Neural Computation (1997)
[6] Fyfe, C.: Two topographic maps for data visualization. Data Mining and Knowledge Discovery 14, 207–224 (2007)
[7] Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of Artificial Intelligence Research 4, 237–285 (1996)

[8] Kohonen, T.: Self-Organising Maps. Springer, Heidelberg (1995)
[9] Likas, A.: A reinforcement learning approach to on-line clustering. Neural Computation (2000)
[10] Sutton, R.S., Barto, A.G.: Reinforcement Learning: an Introduction. MIT Press, Cambridge (1998)
[11] Williams, R.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning 8, 229–256 (1992)
[12] Williams, R.J., Pong, J.: Function optimization using connectionist reinforcement learning networks. Connection Science 3, 241–268 (1991)

Mining Frequent Itemsets in Large Data Warehouses: A Novel Approach Proposed for Sparse Data Sets

S.M. Fakhrahmad[1], M. Zolghadri Jahromi[2], and M.H. Sadreddini[2]

[1] Faculty member in Department of Computer Eng., Islamic Azad University of Shiraz and PhD student in Shiraz University, Shiraz, Iran
[2] Department of Computer Science &Engineering, Shiraz University, Shiraz, Iran
mfakhrahmad@cse.shirazu.ac.ir,{zjahromi,sadredin}@shirazu.ac.ir

Abstract. Proposing efficient techniques for discovery of useful information and valuable knowledge from very large databases and data warehouses has attracted the attention of many researchers in the field of data mining. The well-known Association Rule Mining (ARM) algorithm, Apriori, searches for frequent itemsets (i.e., set of items with an acceptable support) by scanning the whole database repeatedly to count the frequency of each candidate itemset. Most of the methods proposed to improve the efficiency of the Apriori algorithm attempt to count the frequency of each itemset without re-scanning the database. However, these methods rarely propose any solution to reduce the complexity of the inevitable enumerations that are inherited within the problem. In this paper, we propose a new algorithm for mining frequent itemsets and also association rules. The algorithm computes the frequency of itemsets in an efficient manner. Only a single scan of the database is required in this algorithm. The data is encoded into a compressed form and stored in main memory within a suitable data structure. The proposed algorithm works in an iterative manner, and in each iteration, the time required to measure the frequency of an itemset is reduced further (i.e., checking the frequency of n-dimensional candidate itemsets is much faster than those of n-1 dimensions). The efficiency of our algorithm is evaluated using artificial and real-life datasets. Experimental results indicate that our algorithm is more efficient than existing algorithms.

Keywords: Data Mining, Frequent Itemset, Association Rule Mining, Transactional Database, Logical Operations.

1 Introduction

Mining association rules (ARs) is a popular and well researched field in data mining for discovery of interesting relations between items in large databases and transaction warehouses. Their most popular applications include market basket data analysis, cross-marketing, catalog design, information retrieval, clustering and classification [1,2,3].

ARs are represented in the general form of $X \rightarrow Y$ and imply a co-occurrence relation between X and Y, where X and Y are two sets of items (called itemsets). X

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 517–526, 2007.

and Y are called antecedent (left-hand-side or LHS) and consequent (right-hand-side or RHS) of the rule, respectively. Many evaluation measures are defined to select interesting rules from the set of all possible candidate rules. The most widely used measures for this purpose are minimum thresholds on support and confidence.

In most cases, we are just interested in ARs involving itemsets that appear frequently. For example, we cannot run a good marketing strategy involving items that are infrequently bought. Thus, most of mining methods assume that we only care about set of items that appear together in at least an acceptable percentage of the transactions, i.e., the minimum *support* threshold. The *support* of an itemset X is defined as the proportion of transactions in the data set containing X. The term *frequent itemset* is used for itemsets with high value of support.

The confidence of a rule $X \rightarrow Y$ is defined as $supp(X \cap Y)/supp(X)$, i.e., a fraction of transactions containing X, which contain Y as well. ARs must satisfy a minimum degree of support and confidence at the same time. In this paper, we use the short terms *MinSupp* and *MinConf* for minimum support and minimum confidence thresholds, respectively.

Most association rule mining (ARM) algorithms generate association rules in two steps: (1) Mining all frequent itemsets, and (2) generating all rules using these itemsets. The base of such algorithms is the fact that any subset of a frequent itemset must also be frequent, and that both the LHS and the RHS of a frequent rule must also be frequent. Thus, every frequent itemset of length *n* can result in *n* association rules with a single item on the RHS [4,5,6,10,11].

In data mining applications, the data is often too large to fit in main memory. Therefore, the first step of mining ARs is expensive in terms of computation, memory usage and I/O resources. Much of the research effort in this field has been devoted to improving the efficiency of the first step. The main factors used to evaluate these algorithms are the time needed to read data from disk and the number of times each data item has to be read. There are also some approaches, which consider the memory usage as the main factor to be minimized.

Different algorithms use some key principles and tricks to mine frequent itemsets more efficiently. Most of these algorithms try to present a solution to the problem of finding frequent itemsets by reducing the number of times the whole database has to be scanned (i.e., reduce the number of times that the occurrences of itemsets has to be counted). In the literature, many efficient solutions have already been proposed for this problem. However, one key issue, which has rarely been addressed by other researchers in this field is how to compute the frequency of itemsets in an efficient manner. Finding the frequency of an itemset is carried out by counting the number of occurrences of the itemset, which is a very time consuming process due to the large volume of data in data mining applications. In this paper, we focus our attention on how to present an efficient solution for this problem. In our approach, the database is scanned only once and the data is encoded into a compressed form and stored in main memory within a suitable data structure. The proposed algorithm works in an iterative manner, where by each iteration, the time required to measure the frequency of itemsets, is reduced further.

The rest of the paper is organized as follows: Section 2 introduces some efficient ARM algorithms from the literature. In Section 3, we describe our approach and give the detail of the algorithm, *FastARM*. Experimental results using artificial and real-life data sets are presented in Section 4. Finally, we give the conclusion in Section 5.

2 Related Work

Many algorithms that have already been proposed for ARM, use a two step process for generating ARs: 1) mining frequent itemsets, 2) generating ARs from frequent itemsets. The main focus of many of these proposed algorithms is over the first step, where they try to improve the efficiency of the mining process for finding frequent itemsets by reducing the number of read operations from disk, as much as possible. For this purpose, some methods propose solutions to compute the support of some itemsets in order to avoid a number of unnecessary data re-scans. Some others build a special data structures in main memory for this purpose.

Apriori [4] is the most well-known ARM method. The concepts and principles of this method are the basis of many other proposed algorithms. Many improved versions or efficient implementations of the primary Apriori have also been proposed by different researchers. VIPER [5] and ARMOR[1] [6] are two relatively new algorithms which use the Apriori approach, but are much more efficient. VIPER uses a similar data presentation to our proposed method, but it is not efficient because of its need for multiple data re-scans. ARMOR can be considered as the improved version of another efficient algorithm, Oracle [6]. Oracle and ARMOR use a data structure called DAG to optimize their counting operations of itemset occurrences. FP-Growth [7] is another well-known algorithm, which works differently from others. It discovers frequent itemsets without generating any candidate itemset. In this algorithm, the data is read three times from disk and a hash tree structure is built in memory. All frequent itemsets can be found by traversing the hash tree. The main problem of FP-Growth is its heavy utilization of main memory, which is very dependent on the size of database. Running this algorithm for huge data sets is almost impossible due to the limitation of main memory.

The major problems of many ARM methods are their need to read data from the disk iteratively and the time consuming operation of counting the frequency of each itemset [4,5,6,8,10]. The method proposed in this paper attempts to provide a solution for these problems.

3 The Proposed Algorithm

For ease of illustration, we assume the transaction data warehouse as a binary-valued data set having a relational scheme. Each column in this scheme stands for a possible item that can be found in any transaction of the data warehouse and each tuple represents a transaction. Each 0 or 1 value indicates the presence or absence of an itemset in a transaction, respectively. As an example the relation shown in Fig. 1.(b) is the structured form of the data set of Fig. 1.(a), which contains four transactions.

[1] Association Rule Mining based on Oracle.

	Cheese	*Coke*	*Egg*	*Beer*
Cheese, Coke, Egg	1	1	1	0
Cheese, Egg	1	0	1	0
Coke, Cheese, Beer	1	1	0	1
Coke, Beer	0	1	0	1

a) A transactional data set b) A structured presentation of transactions

Fig. 1. A data set containing some transactions of a market basket

3.1 Mining Frequent Itemsets

As the first step of the algorithm, we divide the relation horizontally into some equi-size partitions, each containing k tuples. We comment on choosing the best value for k later in this section. In this relation, each column contains k bits in each partition, thus the group of bits in each column within each partition can be viewed as a k-bit binary code, which is equivalent to a decimal number between 0 and 2^k-1. These decimal numbers are the major elements of our algorithm.

The partitioned relation is scanned just once and the supports of singletons (1-itemsets) are measured to find 1-frequent itemsets. Meanwhile, for each partition, all nonzero decimal values are extracted. For any column of the data set, which represents a frequent singleton, we build a hash table in memory. Each value in this hash table, is a non-zero decimal value extracted from a partition and its access key is the number of that partition (an integer number between 1 and m, where m is the number of partitions). Since we do not insert zero values into the hash tables, then the values recorded in the hash table indicate the regions of the itemset occurrences and limits the search space for the next steps.

	A	*B*	*C*
I	1	0	0
	0	1	0
	0	1	0
	1	1	0
II	0	0	1
	1	0	0
	1	0	0
	1	0	0
III	1	0	1
	1	0	1
	1	0	0
	0	0	1
IV	0	1	1
	0	1	0
	0	0	1
	0	1	0
V	1	0	0
	0	0	1
	1	0	0
	1	0	0
VI	0	1	1
	0	1	0
	0	1	0
	0	1	0

Fig. 2. An Example data set

As an example, consider the data set *r* with 24 transactions, shown in Fig. 2, where A, B and C are three different items. Assume the value of k is set to 4. Thus, the 24 transactions are divided into 6 partitions, each containing 4 tuples. The proposed algorithm with MinSupp set to 0.4, searches for frequent itemsets as follows. The first step involves counting the occurrences of all singletons and constructing the hash tables for the frequent ones. The calculated values for the supports of *A*,*B* and *C* are 0.45, 0.41 and 0.33, respectively.

Here, only the hash tables of *A* and *B* (frequent singletons) are constructed. The hash table for *C* is not constructed because its support is less than the *MinSupp* threshold.

Hash table of *A*:

Keys	I	II	III	V
Values	9	7	14	11

$(9 = (1001)_2, 7 = (0111)_2, 14 = (1110)_2, 11 = (1011)_2)$

Hash table of *B*:

Keys	I	IV	VI
Values	7	13	15

$(7 = (111)_2, 13 = (1101)_2, 15 = (1111)_2)$

The support of a compound itemset such as AB, is easily measured by using the hash tables of its elements (i.e., A and B), instead of scanning the whole database again. In order to calculate the support of a compound itemset, we begin with the smaller hash table (i.e., the one having fewer values). For each key of this hash table, we first verify if it also exists in the other hash table. This verification does not involve any search due to the direct access structure of hash table. If a key exists in both hash tables, then we perform a logical *AND* operation between the the corresponding values related to that key.

The result of the *AND* operation is another integer value, which gives the co-occurrences of A and B in that partition. If the result is zero, it means that there is no simultaneous occurrence of A and B in that partition. We build a similar hash table for the compound itemset, AB, and insert the non-zero integer values resulted from *AND* operations in this table. The size of this hash table is at most equal to the size of the smaller hash table of the two elements. Each number stored in this hash table is equivalent to a binary number, which contains some 1's. The total number of 1's indicates the co-occurrence frequency of A and B. Thus we should just enumerate the total number of 1's for all integer values, instead of scanning the whole database. This measurement can be done using logical Shift Left (*SHL*) or Shift Right (*SHR*) operations over each value and adding up the carry bits until the result is zero (i.e., there is no other 1-bits to be counted).

The *SHR* operation is preferred to *SHL* in cases where the decimal number under investigation has a value less than $2^{k/2}$. The reason is that the equivalent binary codes for such cases do not contain any 1-bit in their left-hand side half, and selecting *SHR* will make enumeration at least two times faster than using *SHL*.

The efficiency of this structure becomes clearer for measuring the support of higher dimensional itemsets. As we proceed to higher dimensional itemsets, the size of hash tables becomes smaller due to new zeros emerging from *AND* operations. These zeros are not inserted into the result hash table.

Let us refer to the above example and continue the mining process. According to the frequent singletons found, the only candidate for 2-frequent itemsets (pairs) is the itemset *AB*. In order to build the hash table of *AB*, each value stored in the hash table of *B* (i.e., the smaller hash table) is selected for logical *AND* operation with a value having the same key stored in *A*'s hash table. The only key present in both hash tables is I, thus the result is a hash table having just one item, as follows.

Hash table of *AB*: (9 & 7 = 1)

Keys	I
Values	1

To measure the support of *AB*, the number of 1's in the value field of this hash table (in the binary form) has to be counted. Since this value is equal to 1 (i.e., 0001), just one *SHR* operation and thus one comparison is enough to count 1's. However, if we had searched all the data to find the co-occurrences of A and B, the number of required comparisons would have been 48 (for reading the value of A and B in all 24 tuples). In general, this improvement is much more apparent for itemsets of higher dimensions.

In a same way, the hash tables of 2-frequent itemsets are then used to mine 3-frequent itemsets and in general, n-frequent itemsets are mined using (n-1)-frequent itemsets. However, we do not use all combinations of frequent itemsets to get (n+1)-frequent itemsets. The Apriori principle [4] is used to avoid verifying useless combinations: "An n-dimensional itemset can be frequent if all of its (n-1)-dimensional subsets are frequent". Thus, for example if AB and AC are two frequent itemsets, their combination is ABC, but we do not combine their hash tables unless the itemset BC is also frequent. If all the n-1 subsets of an n-dimensional itemset are frequent, combining two of them is enough to get the hash table of the itemset.

4 Experimental Results

We conducted two experiments to evaluate the performance of our algorithm, *FastARM* in comparison with four well-known ARM methods, *Apriori*, *VIPER*, *ARMOR* and *FP-Growth*. We implemented the algorithms in C++ on a 3GHz Intel system with 1 GB RAM. We performed experiments on synthetic and real-life data. In all of the experiments we used $k = 32$ for the size of partitions.

4.1 Experiment 1: Synthetic Data

We used 10 data sets each containing $2*10^6$ transactions in this experiment to evaluate the performance of *Apriori*, *VIPER*, *ARMOR* and *FastARM* algorithms. We generated synthetic data sets randomly for 500 distinct items such that the probability of an item being presenct in a transaction is 0.1. In this experiment, we could not

evaluate the performance of *FP-Growth* due to its heavy utilization of main memory. The reason for this is that *FP-Growth* stores the database in a condensed form in main memory (using a data structure called FP-tree).

The results are shown in Fig. 3. The x-axis in these graphs represents the MinSupp threshold values and the y-axis represents the run times of different algorithms. For each specified value of MinSupp, the average run time of each algorithm over 10 data sets is measured and displayed. In this graph, we observe that the execution time of *FastARM* is relatively less than all of the other algorithms. This relative efficiency is more sensible where the value of MinSupp is very low. We also see that there is a considerable improvement in the performance of *FastARM* with respect to both *Apriori* and *VIPER* and also a relative improvement with respect to *ARMOR*.

Table 1 shows the memory consumption of the algorithm throughout each part of the experiment. The values shown in this table represent the amount of memory required for hash tables in each case. Since the hash tables contain the whole information of the primary database (in another format), we can find out the compression rate of the algorithm by comparing these values with the size of database (which is about 100 MB).

Table 1. The memory consumption of the algorithm through each part of the experiment

Probability of 1-bit	Consumed Memory (MB)
0.005	6.8
0.01	11.3
0.05	47.6
0.1	88.7

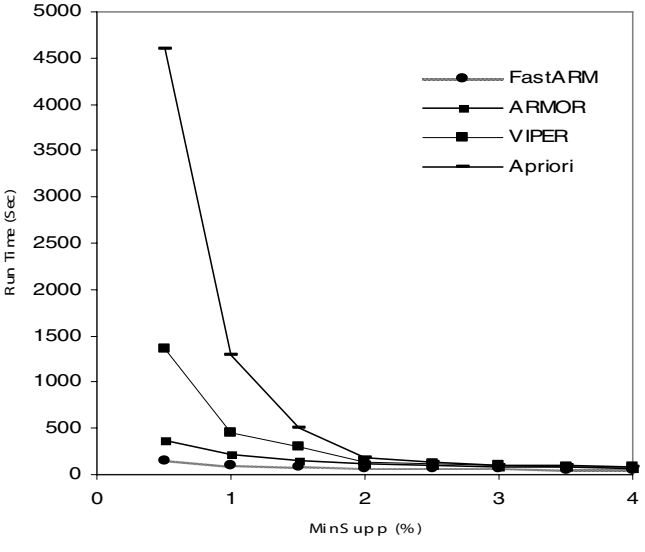
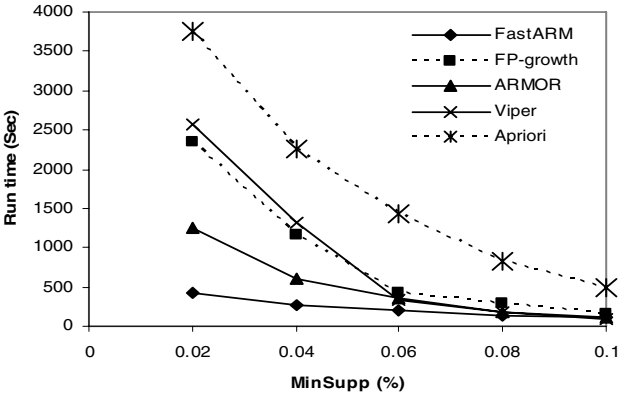
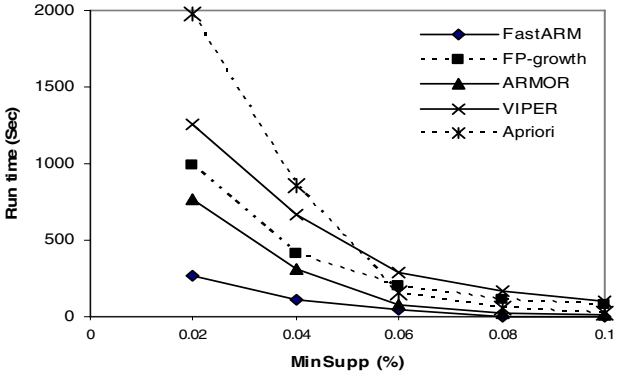


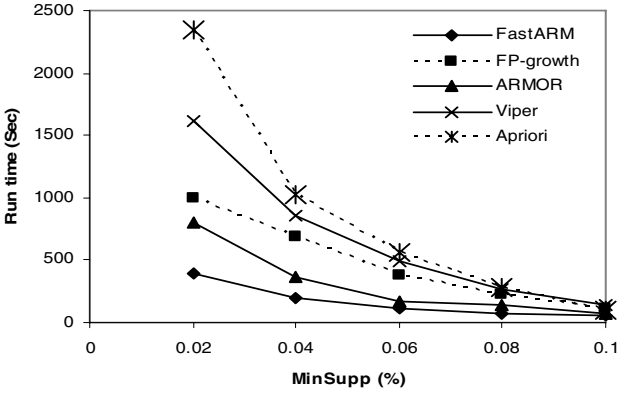
Fig. 3. Performance of different methods on synthetic data for different Minsupp values



a) Running times of various methods on the BMS-POS data set



b) Running times of various methods on the BMS-WebView-1 data set



c) Running times of various methods on the BMS-WebView-2 data set

Fig. 4. Performance of algorithms over some real-life data sets

4.2 Experiment 2: Real Databases

Our second set of experiments involved real data sets extracted from the Frequent Itemset Mining Dataset Repository, namely BMS-POS, BMS-WebView-1 and BMS-WebView-2.

The BMS-POS dataset contains sales data of several years from a large electronics retailer. Since this retailer has so many different products, product categories are used as items. Each transaction in this dataset is a customer's purchase transaction consisting of all product categories purchased at one time. The goal for this dataset is to find associations between product categories purchased by customers in a single visit to the retailer. This data set contains 515,597 transactions and 1,657 distinct items. The BMS-WebView-1 and BMS-WebView-2 datasets contain several months worth of clickstream data from two e-commerce web sites. Each transaction in these data sets is a web session consisting of all the product detail pages viewed in that session. That is, each product detail view is an item. The goal for both of these datasets is to find associations between products viewed by visitors in a single visit to the web site. These two data sets contain 59,602 and 77,512 transactions, respectively (with 497 and 3,340 distinct items).

We set the MinConf threshold value to zero and evaluated the performance of different algorithms using the MinSupp value varying within the range of (0.02%–0.1%). The results of these experiments are shown in Figures 4a–c. We see in these graphs that for lower values of MinSupp, the performance of *FastARM* is significantly better than other methods.

5 Conclusion

In this paper, we proposed an efficient ARM algorithm called the *FastARM* that partitions the data and constructs hash tables to count the frequency of itemsets. Only a single scan of the database is required in this approach and all the necessary information is stored in hash tables. Frequent itemsets are computed by performing the logical AND operations on values from individual hash tables.

We used two experiments on artificial and real-life data sets to evaluate the run time of *FastARM* in comparison with *Apriori*, *FP-Growth, VIPER* and *ARMOR* as four well-known ARM algorithms proposed in the literature. The experiments were conducted to investigate the effect of MinSupp and the database size on the execution time of each algorithm. The results of these experiments clearly indicated that *FastARM* performs better specially for lower values of MinSupp. It should be noticed that as we increase the value of MinSupp, the number of frequent itemsets and generated ARs decreases rapidly. That is why *FastARM* performs similar to the other methods when higher values for MinSupp are used.

References

1. Zamiri, M.J., Rezaei, A.A.: Relationship between blood physiological attributes and carcass characteristics in Iranian fat-tailed sheep. Iranian Journal of Science and Technology, Transactions A 28(A), 97–106 (2004)
2. Ghassem-Sani, G., Halavati, R.: Employing Domain Knowledge to Improve AI Planning Efficiency. Iranian Journal of Science and Technology, Transaction B 29(B1) (2005)

3. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules in Large Databases. In: 20th International Conference on Very Large Data Bases, pp. 487–499 (1994)
4. Shenoy, P., Haritsa, J., Sudarshan, S., Bhalotia, G., Bawa, M., Shah, D.: Turbo-charging vertical mining of large databases. In: ACM SIGMOD Intl. Conf. on Management of Data, vol. 29(2), pp. 22–33. ACM Press, New York (2000)
5. Pudi, V., Haritsa, J.R.: ARMOR: Association Rule Mining based on ORacle. In: ICDM Workshop on Frequent Itemset Mining Implementations, Florida, USA (2003)
6. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent -pattern tree approach. Data Minning and Knowledge Discovery 8(1), 53–87 (2004)
7. Zheng, Z., Kohavi, R., Mason, L.: Real world performance of association rule algorithms. In: KDD. Intl. Conf. on Knowledge Discovery and Data Mining (2001)
8. Michie, D., Spiegelhalter, D.J., Taylor, C.C. (eds.): Machine Learning, Neural and Statistical Classification (STATLOG Project). Herfordshire, Ellis Horwood
9. Pei, J., Han, J., Mao, R.: CLOSET. An efficient algorithm for mining frequent closed itemsets. In: ACM_SIGMOD International Workshop on Data Mining and Knowledge Discovery (2003)
10. Webb, G.I.: OPUS: An efficient admissible algorithm for unordered search. JAIR 3, 431–465 (2004)
11. Webb, G.I.: Efficient search for association rules. In: Sixth ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 99–107. ACM, New York

A Sparse Bayesian Position Weighted Bio-Kernel Network

David C. Trudgian and Zheng Rong Yang

School of Engineering, Computing, and Mathematics,
University of Exeter, Exeter, EX4 4QF, UK
{d.c.trudgian,z.r.yang}@ex.ac.uk

Abstract. The Bio-Basis Function Neural Network (BBFNN) is a successful neural network architecture for peptide classification. However, the selection of a subset of peptides for a parsimonious network structure is always a difficult process. We present a Sparse Bayesian Bio-Kernel Network in which a minimal set of representative peptides can be selected automatically. We also introduce per-residue weighting to the Bio-Kernel to improve accuracy and identify patterns for biological activity. The new network is shown to outperform the original BBFNN on various datasets, covering different biological activities such as as enzymatic and post-translational-modification, and generates simple, interpretable models.

1 Introduction

The Bio-Basis Function Neural Network (BBFNN) [1] is a novel neural architecture that accepts peptide data as input, without requiring the peptides to be numerically encoded. A number of bio-basis functions, which make use of sequence similarity scoring, are used to transform the non-numerical, non-linear input space, with a linear classification stage then being used to make predictions. The BBFNN can be used in both regression and classification modes. Variants of the network are available for use with fixed length peptides, or variable length input sequences, through the use of differing basis functions. This paper is concerned with the classification of fixed length peptides, a common bioinformatics problem.

In the context of peptide classification the BBFNN has been applied to the prediction of cleavage sites, post translational modifications, and aspects of protein structure. Whilst it has been able to produce high quality results for these problems, some aspects of the network merit investigation and improvement. Previous work has investigated the selection of Substitution Matrices used in the Bio-Basis Function, and the production of problem-specific matrices [2]. This paper is concerned with two topics, the selection of the most suitable support peptides from the training data, and the lack of position weightings.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 527–536, 2007.

1.1 Selecting Support Peptides

In the traditional BBFNN model a user must specify the number of basis neurons to be used. Each of these neurons requires a support peptide taken from the training data. Input peptides are compared to the support peptides using the bio-basis function, and the outputs of the function used in the linear classification step of the network. It makes intuitive sense that optimum classification performance would be achieved by using a set of support peptides from the training data which are highly representative of their respective classes. However, the standard BBFNN offers no automatic means of obtaining this set.

The user of the original BBFNN software may manually select a number of support peptides, use a given number of randomly selected peptides, or choose to use all peptides as support peptides. The first case requires careful analysis of the data, and one must be careful not to over-fit the support peptides to the data in hand. The second case is the most frequently used, but can result in relatively large variations in performance depending on the mix of basis peptides selected; selecting the best number of basis neurons to use is also a problem. The final case results in a network that is likely to be over-fitted, and will take a large amount of time to train due to the high complexity. Within all of these methods it has also been common to vary the balance of positive and negative support peptides to try and address class imbalance in the dataset.

It is clear that a method of selecting a set of support peptides that gives good accuracy, is resistant to over-fitting, and results in a parsimonious network that is quick to use, would be of great benefit. We propose using sparse Bayesian learning to accomplish this goal [3].

1.2 Position Weighting

It is reasonable to expect that, in most cases, certain residues will be more important than others in determining the class of a peptide, due to the differing effects on protein structure from each position. Since biological experience shows that all residue positions are not equal in their ability to determine the class of a peptide (e.g. cleavable / non-cleavable), it is reasonable to expect that peptide classification methods that consider specific positions will produce better results than those which do not. Where predictions are made by concentrating on the most informative residues, they are likely to be less affected by noise from unimportant variations at other residue positions. It is therefore reasonable to expect that a position specific method would offer more robust results. However, this is subject to the method not having been over-fitted to certain residues.

Position specific methods in machine learning seek to accommodate this situation, and examine not just the overall composition of a peptide, but consider the amino acids in each position separately. The standard BBFNN is not a position specific method, since the bio-basis function sums the similarity scores for all residue positions without weightings. Methods which use the sparse orthonormal encoding [4] are position specific, each residue has corresponding input nodes and weights. We propose introducing a per-residue weighting to the basis function address this issue.

1.3 Biological Interpretation

The two proposed improvements to the BBFNN are not motivated only by a wish to increase performance, but also to increase the ease with which trained models can be interpreted. Whilst neural network models have been shown to be able to make accurate predictions on biological problems, they are often criticised for being a black box, from which it is hard to extract knowledge. The original BBFNN has a simpler structure than that of multi layer perceptrons, having a single weight layer, and using support peptides and similarity values which can be examined easily. However, by selecting an arbitrary number of support peptides we risk over complicating the model, or excluding potentially interesting support peptides. A method which allows a small model with a set of highly relevant support peptides selected will allow for easier analysis.

By introducing per-residue weights to the basis function we hope additional information useful for biological interpretation will be obtained. On problems where certain positions are known to be more important we hope that the residue weights will allow greater prediction accuracy. On problems where there are no general motifs, we hope that the residue weight information will be useful when interpreting the model, allowing more distinct rules to be identified.

2 Method

Sparse Bayesian Learning, as discussed in [3], is a method to find sparse solutions to models with linearly combined parameters. The BBFNN takes this linear form, where N is the number of basis neurons, w_i is the network weight associated with basis n, y is the model output, and $\Phi(\mathbf{x}, \mathbf{z_i})$ is the value of the basis function applied to input vector $\mathbf{x}$, using support peptide $\mathbf{z_i}$.

$$y = \sum_{i=1}^{N} w_i \Phi(\mathbf{x}, \mathbf{z_i})$$

Since there is a linear combination of our parameters w_n, we can use the method to find a sparse model, where the majority of network weights are close to zero, and can be zeroed, and therefore few basis neurons are required. To carry out sparse Bayesian learning we begin with a network consisting of all possible basis functions, i.e. N is the size of our training dataset. During the learning process network weights, and basis neurons, will be removed. This approach is equivalent to a relevance vector machine (RVM) [3], using the bio-basis function as its kernel function.

We will use the Bernoulli likelihood function, and apply a sigmoid function to the model output y, as is appropriate for two class problems. If $\mathbf{t}$ is the target vector, and l is the number of data points, then the negative log likelihood function of the model is:

$$\mathcal{L} = -\sum_{n=1}^{\ell} \{t_n \log y_n + (1 - t_n) \log(1 - y_n)\}$$

As we will be introducing residue weights which will be learnt using the Bayesian method, we choose to use a kernel function which does not include the normalisation terms, or the exponential operator. The original Bio-Basis Function, given in [1], is such that introducing sparse Bayesian position weighting would generate complicated second derivatives which are inconvenient to work with, and cause the Hessian matrix to be extremely costly to compute. The simplified kernel function, including residue weighting is:

$$\Phi_{ni} = \sum_{d=1}^{D} \theta_d \mathcal{M}_{nid}$$

where θ_d is the weighting for the d^{th} residue in a peptide, D is the number of residues in the peptide, and $\mathcal{M}_{nid}$ is the similarity matrix score between the d^{th} residues of the n^{th} support peptide and i^{th} input peptide. Note that $\theta_d = 1$ when residue weights are not in use.

2.1 Optimisation of Network Weights

For the network weights, a standard Gaussian prior $w_n \sim \mathcal{G}(0, \alpha_n^{w-1})$ is used. The Laplace approximation procedure in [5] is also applied. A Newton-Raphson method optimisation is performed to find the most probable weight vector $\hat{\boldsymbol{w}}$ for the current hyper parameter values $\boldsymbol{\alpha}^w$.

The Hessian matrix, used in the Newton's method optimiser is:

$$H^w = \mathbf{A} + \mathbf{\Phi B \Phi}^T$$

where $\mathbf{A}$ is a diagonal matrix of the hyper-parameters $\boldsymbol{\alpha}_w$, and B is an $l \times l$ diagonal matrix with values $y_n(1 - y_n)$.

This Hessian can be negated and inverted to give the covariance matrix:

$$\Sigma^w = (\mathbf{A} + \mathbf{\Phi B \Phi}^T)^{-1}$$

for a Gaussian approximation to the hyper-parameter posterior, with mean $\hat{\boldsymbol{w}}$.

We will consider our hyper-parameters to be uniformly distributed and therefore only the marginal likelihood must be maximised in order to find $\boldsymbol{\alpha}_w$. We integrate over the negative log of the marginal likelihood with respect to $\mathbf{w}$, and then differentiate with respect to $\boldsymbol{\alpha}_w$ as in [5] to give the update equation:

$$\alpha_i^w = \frac{1 - \alpha_i^w \Sigma_{ii}^w}{\hat{w}_i^2}$$

The training process beings with $\mathbf{w} = 0$, and $\boldsymbol{\alpha}_w = 0.1$. An inner loop implements the Newton method search for the most probable weights given α_w. Once this search has converged, the hyper-parameters are updated in the outer loop. Due to the nature of the sparse Bayesian process, a large number of hyper-parameters will tend to infinity, and their corresponding network weights tend to zero. We prune a weight w_i and its corresponding basis function from the network when $\alpha_i^w > 1 \times 10^{10}$. We continue to update $\mathbf{w}$ and $\boldsymbol{\alpha}_w$ until convergence.

2.2 Optimising Residue Weights

To introduce residue weights we add an additional layer, outside of the hyper-parameter loop to update $\boldsymbol{\theta}$. Note that $\boldsymbol{\theta}$ is a kernel parameter, whereas $\mathbf{w}$ is a network parameter. Again we use the sparse Bayesian approach, resulting in inner and outer loops for optimisation of $\boldsymbol{\theta}$, and the hyper-parameter vector $\boldsymbol{\alpha}^{\theta}$ respectively. These loops sit outside of those for optimisation of the network weights.

Within the inner loop, the most probable residue weights $\hat{\boldsymbol{\theta}}$ given $\boldsymbol{\alpha}^{\theta}$ are again obtained using a Newton's method optimiser. The first derivative of the marginal likelihood $p(\theta|\mathcal{D})$ with respect to θ_d is $-\mathbf{Ze}+\mathbf{A}\boldsymbol{\theta}$, where $\mathbf{A} = \text{diag}(\boldsymbol{\alpha}^{\theta})$, $\mathbf{Z} = (\mathbf{M}_1\mathbf{w}, \mathbf{M}_2\mathbf{w}, \cdots, \mathbf{M}_{\ell}\mathbf{w})$ and

$$\mathbf{M}_n = \begin{pmatrix} \mathcal{M}_{n11} & \mathcal{M}_{n12} & \cdots & \mathcal{M}_{n1\ell} \\ \mathcal{M}_{n21} & \mathcal{M}_{n22} & \cdots & \mathcal{M}_{n2\ell} \\ \vdots & \vdots & \vdots & \vdots \\ \mathcal{M}_{nD1} & \mathcal{M}_{nD2} & \cdots & \mathcal{M}_{nD\ell} \end{pmatrix}$$

The Hessian is then:

$$H^{\theta} = \mathbf{A} + \mathbf{ZBZ}^T$$

$\Delta\theta$ for the Newton-Raphson optimisation in the inner loop is:

$$\Delta\theta = -(\mathbf{Z}^T\mathbf{BZ} + \mathbf{A})^{-1}(\mathbf{A}\theta - \mathbf{Ze})$$

The covariance matrix for the Gaussian approximation is:

$$\Sigma^{\theta} = (\mathbf{A} + \mathbf{ZBZ}^T)^{-1}$$

The marginal likelihood maximisation procedure is applied as previously, to give the hyper-parameter update equation:

$$\alpha_{\theta} = \frac{1 - \alpha_{\theta}\Sigma^{\theta}_{ii}}{\hat{\theta}_i^{\,2}}$$

Initially $\boldsymbol{\theta} = 1$, with $\boldsymbol{\alpha}^{\theta} = 0.1$. At each iteration of the $\boldsymbol{\theta}$ loop, the Φ matrix is recalculated, as with different residue weights the kernel function scores are altered. The network weights $\mathbf{w}$, and weight hyper-parameters $\boldsymbol{\alpha}^{w}$ are updated for each change in $\boldsymbol{\theta}$. No pruning of residue weights is implemented. In practice the hyper-parameters only approach values that would result in pruning on rare occasions.

3 Results and Discussion

3.1 Datasets

We will use three datasets to compare the performance of the new models with the original BBFNN:

GAL - Glycoprotein Linkage Sites. Glycoproteins are an important subset of proteins with a high level of potential pharmacological significance. Carbohydrate groups attached to glycoproteins affect the solubility and thermal stability and are implicated in important biological functions such as controlling uptake of glycoproteins into cells. Chou et al. [6] presented a dataset of 302 9-residue peptides, of which 190 are linkage sites and 112 non-linkage sites.

HIV - HIV-1 Protease Cleavage Sites. During the life-cycle of HIV, precursor polyproteins are cleaved by HIV protease. Disruption of cleavage ability causes non-infectious, imperfect virus replication and is therefore a promising target for anti AIDS drugs. The dataset presented in [7] consists of 8-residue peptides from HIV protease marked as cleavable or non-cleavable. There are 362 peptides of which 114 are positive, cleavage sites and 248 are negative, non cleavable sites.

TCL - T-Cell Epitopes. T-cells are a critical part of the immune response to viral infection. Epitopes are sites on viral proteins that are recognised and bound by the T-cell receptor. The TCL dataset consists of 202 10-residue peptides of which 36 are positive T-cell epitope peptides, the remaining 167 are non-epitope peptides. This data was presented in [8].

3.2 Results

A 5-fold cross validation experiment was used to compare the Sparse Bayesian Bio-kernel Network (SBBKN) with, and without residue weighting, to the original BBFNN. Data was randomised and split into five equal folds. Four folds are used to train a classifier, with the remaining fold then being used to test the model. Each fold is used once for testing, and four times as part of the training set. The entire procedure was repeated twenty times to allow means and standard deviations to be taken for the test statistics. Experiments using the original BBFNN were carried out using 20 bio-basis neurons, with the basis functions randomly selected. The basis functions were not manually selected or balanced as comparison is being made with a new method which does not require manual intervention.

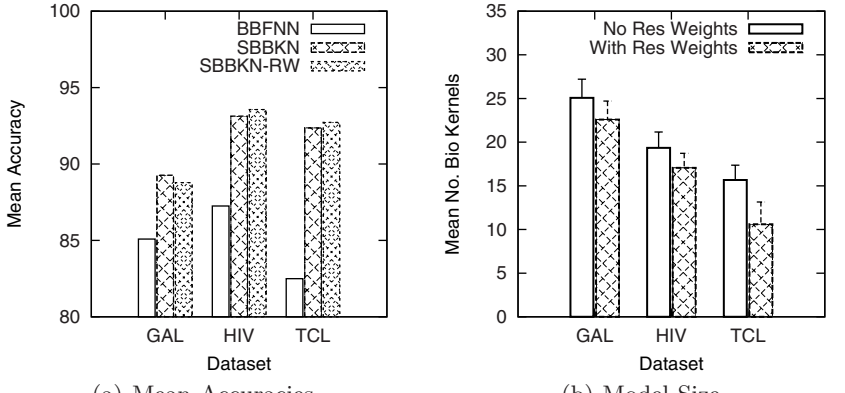
Table 3.2 gives the mean values, and standard deviations in italics, for the test statistics. ACC is the total accuracy, MCC is the value of the Matthew's correlation co-efficient, and AUR is the area under the ROC curve for the model [9], calculated using a 1000 step trapezium method numerical integration.

It can be seen from the ACC column in the table, and more clearly from figure 1(a), that the new techniques outperform the original BBFNN in terms of prediction accuracy on all of the datasets. Standard deviations are slightly larger with the new techniques. The new techniques also outperform the original BBFNN in terms of the Matthews Correlation Coefficient, and mean area under the ROC curves. In the GAL and HIV cases the true negative and positive fractions were both higher than for the original BBFNN. However, on the TCL dataset, which has few positive cases, large improvements in total accuracy and the true negative fraction are accompanied by a smaller decrease in the true positive fraction.

Table 1. Cross Validation Results - Mean & *(Standard Deviation)*

Data	Method	ACC	MCC	AUR
GAL	BBFNN	85.09 *(5.76)*	0.69 *(0.12)*	0.92 *(0.05)*
	SBBKN	89.27 *(8.08)*	0.77 *(0.17)*	0.94 *(0.05)*
	SBBKN-RW	88.78 *(8.27)*	0.76 *(0.18)*	0.94 *(0.06)*
HIV	BBFNN	87.25 *(4.39)*	0.70 *(0.10)*	0.92 *(0.03)*
	SBBKN	93.13 *(5.45)*	0.84 *(0.13)*	0.97 *(0.03)*
	SBBKN-RW	93.56 *(5.46)*	0.85 *(0.13)*	0.97 *(0.03)*
TCL	BBFNN	82.50 *(6.22)*	0.54 *(0.15)*	0.90 *(0.05)*
	SBBKN	92.36 *(6.28)*	0.74 *(0.21)*	0.95 *(0.06)*
	SBBKN-RW	92.72 *(5.11)*	0.75 *(0.17)*	0.96 *(0.05)*

The SBBKN with residue weightings slightly outperforms that without weightings on the HIV and T-Cell data, but falls behind on the glycoprotein linkage data. The small differences suggest that there is little to choose between the methods, except the increased computational cost of residue weightings. However, analysis of the models shows a difference in the number of kernels used to achieve the same performance. Also, the residue weight information may be of biological significance, and therefore useful in further interpretation.



(a) Mean Accuracies (b) Model Size

Fig. 1. Accuracy comparison & model size

The computational cost of the new methods is an increase on that of the standard BBFNN, but is not excessive. On a PC containing an Intel Core 2 Duo processor at 1.7Ghz, the SBBKN without residue weighting takes, on average, 96 seconds to train on 288 peptides from the HIV dataset. With residue weighting the average is 145 seconds. Residue weighting adds relatively little extra

computation time. The majority of kernels will be pruned in the first update cycle, leaving only a small network for the majority of the residue weight update calculations.

3.3 Model Analysis

Having considered raw performance statistics we will examine the models that are produced by the training process. Although it was shown that the performance of the weighted and non-weighted SBBKNs was similar, the weighted SBBKN uses fewer kernels on each problem. Figure 1(b) shows the mean, and standard deviation, of the number of bio-kernels used for each dataset. It can be seen that around three fewer kernels are used on the GAL and HIV sets, with five fewer on the T-Cell data. For the T-Cell data this represents a 32% decrease in the size of the network. Whilst there is increased complexity due to the addition of per-residue weights, this is partially offset by a reduction in the number of bio-kernels required to achieve similar performance.

Further analysis work has been carried out on the residue-weighted models produced for the HIV-1 cleavage site prediction problem. Whilst there have been some questions regarding the use of neural networks on the dataset [10], the availability of motifs in the literature will allow comparison with the residue weights and support peptides used in our trained models. In future we will examine the models for datasets without known strong motifs.

The intuitive first step in the analysis to calculate the mean values of the residue weights, to identify the most important positions in the peptides. However, initial inspection suggested that the weights were highly skewed, precluding the sensible use of the mean and standard deviation in analysis work. Histograms were plotted which confirmed that the distribution is heavily positively skewed. Further investigation showed that in general, where any given θ_d value is high, all other position weights will also be high. When examining the relative importance of residues we are interested in the difference between θ_d values, not their absolute value. With this in mind, we have chosen to scale the values, relative to the highest position weight in each model: $\theta'_d = \theta_d / \theta_{MAX}$. This will allow for meaningful averages to be taken, and would not be necessary if examining a single model. Figure 2 shows the minimum, mean, and median values for the scaled weights.

Whilst there is a large spread of values there appears to be a trend that is consistant between the statistics. Values are high in the P1, P2, P1', and P2' positions. The P2' site is recognised as important for cleavage predictions, with Glutamate (E) or Glutamine (Q) identified as amino acids which indicative of a possible cleavage. At P2, Valine (V) or Alanine (A) are associated with an increased likelihood of the peptide being cleavable [10,11]. The P1 and P1' positions, directly to each side of the cleavage site, are also of importance and similarly mentioned in motifs. The high value of the P4' position was unexpected, since attention is usually paid to the P1-P1' positions [12,7]. However, Lysine (K) at P4' was noted as a contributor to positive predictions in 29% of cases in a model given in [10].

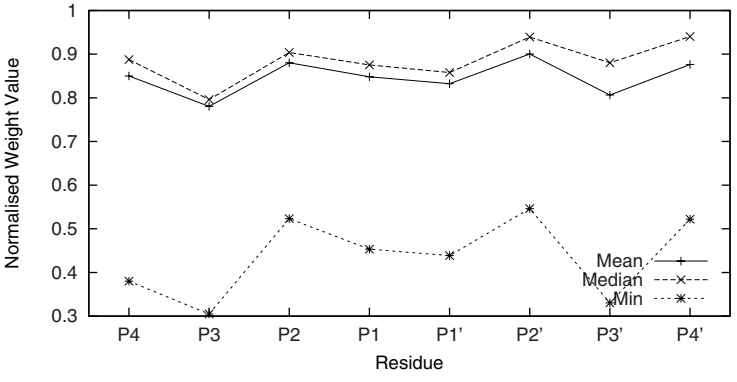


Fig. 2. HIV scaled residue weights

The low importance of the P3 and P3' positions were also noted in [10], where it was found that they could be excluded from models without changing the separability of the data. It is perhaps disappointing that the median value for these residue positions is still fairly high. However, whilst there are only fairly small variations in the statistics between each residue, the trends fit with what might be expected. The majority of residue weight updates take place after pruning of support peptides and network weights has taken place, once the network weights are tuned for high performance on with a small set of support peptides. It may be the case that allowing for a cycle of residue weight updates before any pruning of network weights takes place would allow the values to move further; with less tuned network weights, changes to the residue weights would likely cause larger changes in error, in turn causing larger updates. Further investigation would be useful.

One aspect of the motivation for the use of the sparse Bayesian approach was to identify a parsimonious model, with a minimal number of support peptides. For the HIV case there is a mean of 17.07 support peptides per model, of which 53% are positive, i.e. actual cleavage sites. Within the 5 fold, 20 repeat cross validation procedure, each peptide occurs in the training data for 80 models. Therefore the most common support peptide, `DAINTEFK`, observed in 68 models, occurred in 85% of possible cases. The 30 most common support peptides account for 1027 out of the 1707 selections across 100 models, i.e. 60% of support peptides are selected from 8% of the training data. The fact that this small number of peptides are commonly selected across randomly populated training folds seems to indicate that they are representative across all of the data, and that over-fitting to training sets has been eliminated by the sparse Bayesian method.

4 Conclusions

We have introduced a sparse Bayesian bio-kernel network, based on the Bio-Basis Function Neural Network. The new method produces models which have been

shown to make accurate predictions on three datasets, using only a small number of support peptides. In addition we have introduced a variant that includes per-residue weights in the basis function. With the added residue weights, slightly better classification accuracies are achieved using fewer support peptides. Analysis of the models produced for HIV-1 protease cleavage site prediction identifies patterns which agree with previous literature.

References

1. Thomson, R., Hodgman, T.C., Yang, Z.R., Doyle, A.K.: Characterising proteolytic cleavage site activity using bio-basis function neural networks. Bioinformatics 19, 1741–1747 (2003)
2. Trudgian, D.C., Yang, Z.R.: Substitution Matrix Optimisation for Peptide Classification. Lecture Notes In Computer Science, vol. 4447, pp. 291–300. Springer, Heidelberg (2007)
3. Tipping, M.E.: Sparse Bayesian Learning and the Relevance Vector Machine. J. Machine Learning Res. 1, 211–244 (2001)
4. Qian, N., Sejnowski, T.J.: Predicting the secondary structure of globular proteins using neural network models. J. Mol. Biol. 202, 865–884 (1988)
5. MacKay, D.J.C.: Bayesian Interpolation. Neural Computation 4(3), 415–447 (1992)
6. Chou, K., Zhang, C., Kezdy, F.J., Poorman, R.A.: A Vector Projection Method for Predicting the Specificity of GalNAc-Transferase. Proteins 21, 118–126 (1995)
7. Cai, Y., Chou, K.: Artificial neural network model for predicting HIV protease cleavage sites in protein. Adv. in Eng. Software 29(2), 119–128 (1998)
8. Zhao, Y., Pinilla, C., Valmori, D., Martin, R., Simon, R.: Application of support vector machines for T-cell epitopes prediction. Bioinformatics 19(15), 1978–1984 (2003)
9. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143(1), 29–36 (1982)
10. Rognvaldsson, T., You, L.: Why neural networks should not be used for HIV-1 protease cleavage site prediction. Bioinformatics 20(11), 1702–1709 (2004)
11. You, L., Garwicz, D., Rognvaldsson, T.: Comprehensive Bioinformatic Analysis of the Specificity of Human Immunodeficiency Virus Type 1 Protease. J. Virology 79(19), 12477–12486 (2005)
12. Narayanan, A., Wu, X., Yang, Z.: Mining viral protease data to extract cleavage knowledge. Bioinformatics 18, S5–S13 (2002)

Square Penalty Support Vector Regression

Álvaro Barbero, Jorge López, and José R. Dorronsoro*

Dpto. de Ingeniería Informática and Instituto de Ingeniería del Conocimiento
Universidad Autónoma de Madrid, 28049 Madrid, Spain

Abstract. Support Vector Regression (SVR) is usually pursued using the ϵ–insensitive loss function while, alternatively, the initial regression problem can be reduced to a properly defined classification one. In either case, slack variables have to be introduced in practical interesting problems, the usual choice being the consideration of linear penalties for them. In this work we shall discuss the solution of an SVR problem recasting it first as a classification problem and working with square penalties. Besides a general theoretical discussion, we shall also derive some consequences for regression problems of the coefficient structure of the resulting SVMs and illustrate the procedure on some standard problems widely used as benchmarks and also over a wind energy forecasting problem.

1 Introduction

Standard ϵ–insensitive SVR [9,12] seeks to minimize $\|W\|^2$ subject to the restrictions $W \cdot X^i + b - (y^i - \epsilon) \geq 0$, $W \cdot X^i + b - (y^i + \epsilon) \leq 0$. If it exists, the solution of this problem, that we shall refer to as ϵ–SVR, defines what is usually called a hard ϵ tube. However, in practical problems, hard tubes have to be replaced by soft ones, where besides ϵ insensitivity, extra slack terms have to be introduced. More precisely, the previous restrictions become now

$$W \cdot X^i + b - (y^i - \epsilon) + \xi_i \geq 0, \quad W \cdot X^i + b - (y^i + \epsilon) - \mu_j \leq 0, \tag{1}$$

and the function to be minimized is now $J_p(W, b, \xi, \mu) = \|W\|^2 + C \sum (\xi_i^p + \mu_j^p)$ for some $p \geq 1$ and where C is a properly chosen penalty factor. Obviously, minimizing $J_p(W, b, \xi, \mu)$ is equivalent to minimizing

$$\sum_i \left[y^i - f(X^i, W, b) \right]_\epsilon^p + \lambda \|W\|^2,$$

for $\lambda = 1/C$ and where $f(X, W, b) = W \cdot X + b$ and $[z]_\epsilon = \max(0, |z| - \epsilon)$. Thus, the problem that soft ϵ–insensitive SVR solves can be seen as a modelling problem where errors are measured in terms of the $[\cdot]_\epsilon^p$ function and a regularization term $\lambda \|W\|^2$ is added. As it is the case in standard soft margin SVMs, the usual choice in SVR is to take $p = 1$; however, in this work we shall take $p = 2$, which is

* All authors have been partially supported by Spain's TIN 2004–07676.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 537–546, 2007.

also a frequent election in SVM training. For either choice, rather than minimize the criterion $J_p(W, b, \xi, \mu)$ one defines a certain dual problem with a quadratic programming structure and that can be solved by standard packages or simply by gradient ascent on the dual function (see [13] for the linear penalty case). Here, however, we will take a different point of view. We note first that SVR can be transformed in a classification problem [2]. More precisely, if an ϵ hard tube exists, shifting the y_i values by $\pm\epsilon$ we obtain subsets $\mathcal{D}^+ = \{(X^i, y^i + \epsilon)\}$ and $\mathcal{D}^- = \{(X^i, y^i - \epsilon)\}$ that are linearly separable, and the ϵ–insensitive SVR problem can then be recast as that of minimizing the quantity $\|W\|^2 + \delta^2$ subject to the conditions

$$W \cdot X^i + b - \delta(y^i - \tilde{\epsilon}) \geq 1, \;\; W \cdot X^i + b - \delta(y^i + \tilde{\epsilon}) \leq -1, \tag{2}$$

where $\epsilon = \tilde{\epsilon} - 1/\delta$; we shall call this problem $\tilde{\epsilon}$–SVC. Its dual function is then

$$\Theta(\alpha, \beta) = -\frac{1}{2}\|X_\alpha - X_\beta\|^2 - \frac{1}{2}\left(\tilde{\epsilon}\sum(\alpha_i + \beta_i) - \sum(\alpha_i - \beta_i)y^i\right)^2 + \sum(\alpha_i + \beta_i) \tag{3}$$

subject to the restrictions $\alpha_i, \beta_j \geq 0$ and $\sum \alpha_i = \sum \beta_j$, and where $X_\alpha = \sum \alpha_i X^i$, $X_\beta = \sum \beta_j X^j$. We can get a more compact formulation of $\tilde{\epsilon}$–SVC writing points in $\mathcal{D}_\pm$ as $X^i_+ = (X^i, y^i + \tilde{\epsilon})$, $X^j_- = (X^j, y^j - \tilde{\epsilon})$ and the weight vector as $\widetilde{W} = (W, -\delta)$. Then $\tilde{\epsilon}$–SVC can be stated as minimizing $\|\widetilde{W}\|^2$ subject to $\widetilde{W} \cdot X^i_- + b \geq 1$, $\widetilde{W} \cdot X^j_+ + b \leq -1$. A possible way to solve it is to find [1] the closest points X^*_+ and X^*_- in the convex hulls $C(\mathcal{D}_+)$, $C(\mathcal{D}_-)$ of $\mathcal{D}_+$ and $\mathcal{D}_-$. We shall call this third convex hull formulation CH–SVM.

Our approach to solve square penalty ϵ–SVR will be based on the solution of CH–SVM. More precisely, we will show in section 2 the equivalence for the hard margin setting of $\tilde{\epsilon}$–SVC and CH–SVM, and how to reduce ϵ–SVR to $\tilde{\epsilon}$–SVC; in particular, we will see how their solutions are related. An advantage of using square penalties is that hard and soft SVMs can be treated formally in the same way. In section 3 we will recall how this is done and, turning our attention to ϵ–SVR, our main contribution will be Proposition 3, where we show that the coefficient structure of the optimal solution of soft CH–SVM can be seen as defining a certain regression tube, slightly larger than the original ϵ one. Patterns correctly classified by the CH–SVM solution will fall inside it, while not correctly classified patterns will fall outside. In section 4 we will illustrate the application of square penalty SVR to some publicly available regression datasets as well as in a wind energy prediction problem. The paper will end with a short discussion and some conclusions.

2 Hard Margin SV Regression and Classification

It is well known [9] that the optimal ϵ–SVR solution weight $\widehat{W}$ can be written as $\widehat{W} = X_{\widehat{\alpha}} - X_{\widehat{\beta}}$, with $\widehat{\alpha}, \widehat{\beta}$ the optimal dual solutions. Moreover, the Karush–Kuhn–Tucker (KKT) conditions verified by the optimal $\widehat{W}, \widehat{b}, \widehat{\alpha}, \widehat{\beta}$ imply that if

at some i we have, say, $\widehat{\alpha}_i > 0$, then $\widehat{W} \cdot X^i + \widehat{b} - (y^i - \epsilon) = 0$ and, therefore, $\widehat{b} = (y^i - \epsilon) - \widehat{W} \cdot X^i = 0$. Similarly, the optimal $\tilde{\epsilon}$–SVC solution weight W^o can be written as $W^o = X_{\alpha^o} - X_{\beta^o}$ [2], with α^o, β^o now the optimum solutions of the corresponding dual problem; moreover, the optimal δ^o value is given by

$$\begin{aligned}\delta^o &= \sum \left\{\beta_i^o \left(y^i + \tilde{\epsilon}\right) - \alpha_i^o \left(y^i - \tilde{\epsilon}\right)\right\} \\ &= \tilde{\epsilon} \sum \left(\alpha_i^o + \beta_i^o\right) - \sum \left(\alpha_i^o - \beta_i^o\right) y^i. \end{aligned} \tag{4}$$

Finally, the KKT conditions are in this case

$$\begin{aligned}\alpha_i^o > 0 \Rightarrow W^o \cdot X^i - \delta^o(y^i - \tilde{\epsilon}) + b^o = 1, \\ \beta_j^o > 0 \Rightarrow W^o \cdot X^j - \delta^o(y^i + \tilde{\epsilon}) + b^o = -1,\end{aligned}$$

and just as before, the optimal b^o can be obtained as, say, $b^o = 1 + \delta^o(y^i - \tilde{\epsilon}) - W^o \cdot X^i$ if $\alpha_i^o > 0$. The following proposition relates the optimal solutions of ϵ–SVR and $\tilde{\epsilon}$–SVC. Although essentially known in the literature, we shall give here its proof for a lack of a proper reference.

Proposition 1. *Assume $\tilde{\epsilon}$ to be such that the shifted classes $\mathcal{D}_+$, $\mathcal{D}_-$ are linearly separable and let W^o, δ^o, b^o and $\widehat{W}, \widehat{b}$ be the optimal solution of $\tilde{\epsilon}$–SVC and ϵ–SVR respectively. We then have $W^o = \delta^o \widehat{W}$, $b^o = \delta^o \widehat{b}$ and $\epsilon = \tilde{\epsilon} - 1/\delta^o$.*

Proof. If W^o, δ^o, b^o is the optimal solution of $\tilde{\epsilon}$–SVC, it easily follows from (2) that $\tilde{\epsilon} \geq 1/\delta^o$. If $\tilde{\epsilon}\delta^o = 1$, the restrictions in (2) would become $W^o \cdot X^i - \delta^o y^i + b^o \geq 0$, $W^o \cdot X^i - \delta^o y^i + b^o \leq 0$ for all i. This would imply $\frac{W^o}{\delta^o} \cdot X^i + \frac{b^o}{\delta^o} = y^i$ for all i, i.e., we would have a perfect fit at all points, an unusual circumstance not likely to happen; hence, we will assume $\tilde{\epsilon} > 1/\delta^o$. Then $\tilde{W} = W^o/\delta^o$ and $\tilde{b} = b^o/\delta^o$ is a feasible solution of ϵ–SVR with $\epsilon = \tilde{\epsilon} - 1/\delta^o$. As a consequence, $\|\widehat{W}\| \leq \|\tilde{W}\| = \|W^o\|/\delta^o$.

On the other hand and as we have just mentioned (see also [9]), the optimal solution $\widehat{W}$ of ϵ–SVR can be written as $\widehat{W} = \sum_i \widehat{\alpha}_i X^i - \sum \widehat{\beta}_j X^j$, with $\sum_i \widehat{\alpha}_i = \sum \widehat{\beta}_j$. The KKT conditions imply that at an $\widehat{\alpha}_i > 0$ SV X^i we have $\widehat{W} \cdot X^i + \widehat{b} - (y^i - \epsilon) = 0$, while at a $\widehat{\beta}_j > 0$ SV X^j we have $\widehat{W} \cdot X^j + \widehat{b} - (y^j + \epsilon) = 0$. Writing now $\epsilon = \tilde{\epsilon} - 1/\delta^o$, it follows that

$$\widehat{W} \cdot X^j + \widehat{b} - (y^j - \tilde{\epsilon}) = \frac{1}{\delta^o}; \;\; \widehat{W} \cdot X^j + \widehat{b} - (y^j + \tilde{\epsilon}) = \frac{-1}{\delta^o},$$

and, therefore,

$$\delta^o \widehat{W} \cdot x^j + \delta^o \widehat{b} - \delta^o(y^j - \tilde{\epsilon}) = 1; \;\; \delta^o \widehat{W} \cdot x^j + \delta^o \widehat{b} - \delta^o(y^j + \tilde{\epsilon}) = -1.$$

Thus, $(W' = \delta^o \widehat{W}, \delta^o, b' = \delta^o \widehat{b})$ is a feasible solution of $\tilde{\epsilon}$–SVC and, hence, $\delta^o \|\widehat{W}\| = \|W'\| \geq \|W^o\|$. By the uniqueness [3] of the SVM solutions, it follows that $\widehat{W} = W^o/\delta^o$ and the other equalities are then immediate. □

Turning our attention to the relationship between $\tilde{\epsilon}$–SVC and CH–SVM, recall that writing $X_+ = (X, y + \tilde{\epsilon})$, $X_- = (X, y - \tilde{\epsilon})$ and $\widetilde{W} = (W, -\delta)$, $\tilde{\epsilon}$–SVC minimizes $\|\widetilde{W}\|^2$ subject to $\widetilde{W} \cdot X_-^i + b \geq 1$, $\widetilde{W} \cdot X_+^j + b \leq -1$. As mentioned before, the optimal solution of CH–SVM is given by the closest points X_+^* and X_-^* in the convex hulls $C(\mathcal{D}_\pm)$ of $\mathcal{D}_\pm$ (see [1] for more details). They verify therefore that $X_-^* = \sum \alpha_i^* X_-^i$ and $X_+^* = \sum \beta_j^* X_+^j$, with $\sum \alpha_i^* = \sum \beta_j^* = 1$ and define an optimal vector $\widetilde{W}^*$ and bias b^* as

$$\widetilde{W}^* = X_-^* - X_+^*,\ b^* = \frac{1}{2}\left(\|X_-^*\|^2 - \|X_+^*\|^2\right).$$

Moreover, the maximum margin m^* is given by $m^* = \|\widetilde{W}^*\|/2$. The following proposition is also known [1] and proved using the KKT conditions of each problem.

Proposition 2. *The optimal solution $\widetilde{W}^*$ of CH–SVM is related to the optimal $\widetilde{W}^o$ of $\tilde{\epsilon}$–SVC as*

$$\widetilde{W}^o = \frac{2}{\|\widetilde{W}^*\|^2}\widetilde{W}^* = \frac{1}{m^*}\tilde{W}^*,$$

or, equivalently, $\widetilde{W}^* = 2\widetilde{W}^o/\|\widetilde{W}^o\|^2$. Moreover, $W^o = 2W^*/\|W^*\|^2$, $b^o = 2b^*/\|W^*\|^2$ and $\delta^o = 2\delta^*/\|W^*\|^2$.

CH–SVM is somewhat different formally from ϵ–SVR and $\tilde{\epsilon}$–SVC and although still solvable using quadratic programming tools, it lends itself to algorithms quite different from those standard in SVMs. A good choice is the Schlesinger–Kozinec (SK) algorithm [6]. The starting observation is that for any potential weight $\widetilde{W} = X_- - X_+$, with $X_\pm \in C(\mathcal{D}_\pm)$, its margin $m(W)$ verifies $m(W) \leq \|W\|/2$. Moreover, if $\widetilde{W}^* = X_-^* - X_+^*$ is the optimal weight, we have $m(\widetilde{W}) \leq m(\widetilde{W}^*) = \|\widetilde{W}^*\|/2 \leq \|\widetilde{W}\|/2$. Thus setting $g(\widetilde{W}) = \|\widetilde{W}\|/2 - m(\widetilde{W})$, we have $0 = g(\widetilde{W}^*) \leq g(\widetilde{W})$. The SK algorithm iteratively constructs approximations $\widetilde{W}^t$ to W^* by convex updates $\widetilde{W}^t = (1-\lambda^t)\widetilde{W}^{t-1} + \lambda^t \widetilde{X}_\pm^t$, where λ^t and $\widetilde{X}_\pm^t \in C(\mathcal{D}_\pm)$ are chosen to ensure that $\widetilde{W}^t < \widetilde{W}^{t-1}$ and that approximately (although not true for all iterations) $m(\widetilde{W}^t) < m(\widetilde{W}^{t-1})$ (see [6] for more details). We shall use the SK algorithm in our square penalty experiments.

3 Square Penalty SV Regression and Classification

Recall that square penalty ϵ–SVR seeks to minimize $\|W\|^2 + C\sum(\xi_i^2 + \mu_j^2)$ subject to the restrictions $W \cdot X^i + b - (y^i - \epsilon) + \xi_i \geq 0$, $W \cdot X^i + b - (y^i + \epsilon) - \mu_j \leq 0$. It can be reduced to a hard ϵ–SVR problem by extending the W and X vectors adding $2N$ extra coordinates to them, with N the sample size, and defining

$$\overline{W} = (W, \sqrt{C}\xi_1, \ldots, \cdot\sqrt{C}\xi_N, \sqrt{C}\mu_1, \ldots, \sqrt{C}\mu_N),$$

$$\overline{X}^i_- = (X^i, 0, \ldots, \tfrac{1}{\sqrt{C}}, \ldots, 0, 0, \ldots, 0), \;\; \overline{X}^j_+ = (X^j, 0, \ldots, 0, 0, \ldots, \tfrac{-1}{\sqrt{C}}, \ldots, 0),$$

where the final non–zero coordinate of $\overline{X}^i_-$ is the extra i–th one and the final non–zero coordinate of $\overline{X}^j_+$ is the extra $(N+j)$–th one. We then have $\|W\|^2 + C\sum(\xi_i^2 + \mu_j^2) = \|\overline{W}\|^2$ and the previous restrictions become $\overline{W} \cdot \overline{X}^i_- + b - (y^i - \epsilon) \geq 0$, $\overline{W} \cdot \overline{X}^j_+ + b - (y^i + \epsilon) \leq 0$.

We can similarly reduce square penalty $\tilde{\epsilon}$–SVC to a hard $\tilde{\epsilon}$–SVC problem. Keeping the previously used $X_\pm = (X, y \pm \tilde{\epsilon})$ and $\widetilde{W} = (W, -\delta)$ notation, we consider now the extended weight and vectors

$$\overline{W} = (\widetilde{W}, \sqrt{C}\xi_1, \ldots, \sqrt{C}\xi_N, \sqrt{C}\mu_1, \ldots, \sqrt{C}\mu_N),$$

$$\overline{X}^i_- = (X^i_-, 0, \ldots, \frac{1}{\sqrt{C}}, \ldots, 0, 0, \ldots, 0), \; \overline{X}^j_+ = (X^j_+, 0, \ldots, 0, 0, \ldots, \frac{-1}{\sqrt{C}}, \ldots, 0),$$

for which we have again $\|W\|^2 + \delta^2 + C\sum(\xi_i^2 + \mu_j^2) = \|\overline{W}\|^2$ and the restrictions $\overline{W} \cdot \overline{X}^i_- + b \geq 1$, $\overline{W} \cdot \overline{X}^j_+ + b \leq -1$. Solving the CH–SVM version of $\tilde{\epsilon}$–SVC will give the optimal extended weight $\overline{W}^*$ as $\overline{W}^* = \overline{X}^*_- - \overline{X}^*_+$, with $\overline{X}^*_- = \sum \alpha_i^* \overline{X}^i_-$ and $\overline{X}^*_+ = \sum \beta_j^* \overline{X}^j_+$. In particular we will have

$$\begin{aligned}\overline{W}^* &= (\widetilde{W}^*, \sqrt{C}\xi_1^*, \ldots, \sqrt{C}\xi_N^*, \sqrt{C}\mu_1^*, \ldots, \sqrt{C}\mu_N^*) \\ &= (\sum \alpha_i^* X^i_- - \sum \beta_j^* X^j_+, \frac{\alpha_1^*}{\sqrt{C}}, \ldots, \frac{\alpha_N^*}{\sqrt{C}}, \frac{\beta_1^*}{\sqrt{C}}, \ldots, \frac{\beta_N^*}{\sqrt{C}}),\end{aligned}$$

and, therefore, margin slacks and SV coefficient are directly related as $C\xi_i^* = \alpha_i^*$, $C\mu_j^* = \beta_j^*$. Moreover, as we will shall see next, the size of the optimal α_i^*, β_j^* coefficients determine the tube in which patterns X^j_-, X^j_+ will fall.

Proposition 3. *Set* $\Lambda^* = C\|\overline{W}^*\|^2 = \sum \left\{(\alpha_i^*)^2 + (\beta_i^*)^2\right\} + C\left(\|W^*\|^2 + (\delta^*)^2\right)$. *Then a pattern* (X^i, y^i) *will fall inside the* $\tilde{\epsilon}$ *tube if and only if* $\alpha_i^* < \Lambda^*/2$ *and* $\beta_i^* < \Lambda^*/2$.

Proof. We will bring the extended CH–SVM solution $\overline{W}^*$ back to the ϵ–SVR one retracing the steps already mentioned in the penalty–free case. We go first from $\overline{W}^*$ to the optimal solution $\overline{W}^o$ of $\tilde{\epsilon}$–SVC as $\overline{W}^o = 2\overline{W}^*/\|\overline{W}^*\|^2$. As a consequence, the optimal $\tilde{\epsilon}$–SVC slack variables verify

$$\xi_i^o = \frac{2}{\|\overline{W}^*\|^2}\,\xi_i^* = \frac{2}{\Lambda^*}\,\alpha_i^*, \;\; \mu_j^o = \frac{2}{\|\overline{W}^*\|^2}\,\mu_j^* = \frac{2}{\Lambda^*}\,\beta_j^*. \tag{5}$$

Now, since we have $\delta^o = 2\delta^*/\|\overline{W}^*\|^2 = 2C\delta^*/\Lambda^*$, proposition 1 and (5) imply that the ϵ–SVR slacks are

$$\begin{aligned}\hat{\xi}_i &= \frac{1}{\delta^o}\xi_i^o = \frac{\Lambda^*}{2C\delta^*}\frac{2}{\Lambda^*}\,\alpha_i^* = \frac{1}{C\delta^*}\alpha_i^*, \\ \hat{\mu}_j &= \frac{1}{\delta^o}\mu_j^o = \frac{\Lambda^*}{2C\delta^*}\frac{2}{\Lambda^*}\,\beta_j^* = \frac{1}{C\delta^*}\beta_j^*.\end{aligned}$$

Table 1. SVM parameters used. For the `flare` and `building` problems only the parameters for the first output are shown.

	Linear Penalty			Square Penalty		
Problem	C	σ	ϵ	C	σ	$\tilde{\epsilon}$
flare1	160	250	0.0025	0.125	10	0.04
flare2	30	150	0.001	0.5	12	0.06
flare3	40	175	0.001	3	40	0.05
building1	0.6	125	0.01	0.3	25	0.4
building2	3.2	8	0.01	0.8	6	0.6
building3	6.3	6.5	0.01	0.8	5	0.6
wind power	0.4	32	0.08	5	40	0.2

Furthermore, since $\epsilon = \tilde{\epsilon} - 1/\delta^o$, we have

$$\epsilon + \hat{\xi}_i = \tilde{\epsilon} - \frac{\Lambda^*}{2C\delta^*} + \frac{\alpha_i^*}{C\delta^*} = \tilde{\epsilon} - \frac{1}{C\delta^*}\left(\frac{\Lambda^*}{2} - \alpha_i^*\right),$$
$$\epsilon + \hat{\mu}_j = \tilde{\epsilon} - \frac{\Lambda^*}{2C\delta^*} + \frac{\beta_j^*}{C\delta^*} = \tilde{\epsilon} - \frac{1}{C\delta^*}\left(\frac{\Lambda^*}{2} - \beta_j^*\right).$$

Since $\alpha_i^o = 0$ if and only if $\xi_i^o = 0$, all the regression patterns (X^i, y^i) for which $\alpha_i^o = \beta_i^o = 0$ will be inside an $\hat{\epsilon}$ tube with $\hat{\epsilon} = \tilde{\epsilon} - \Lambda^*/2C\delta^*$. Next, since α_i^* and β_i^* cannot be simultaneously nonzero, patterns $(X^i, y^i \pm \tilde{\epsilon})$ for which either coefficient is $< \Lambda^*/2$ result in regression patterns (X^i, y^i) inside the $\tilde{\epsilon}$–hard tube. On the other hand if, say, $\alpha_i^* > \Lambda^*/2$, the KKT conditions now imply

$$\widehat{W} \cdot X^i + b - y^i = \tilde{\epsilon} + \frac{1}{C\delta^*}\left(\alpha_i^* - \frac{\Lambda^*}{2}\right) > \tilde{\epsilon};$$

that is, (X^i, y^i) will fall outside the $\tilde{\epsilon}$–hard tube, and the same will happen with those (X^j, y^j) for which $\beta_j^* > \Lambda^*/2$. □

We will illustrate next square penalty SVR over several regression problems, comparing its performance to that of linear penalty SVR and of multilayer perceptrons.

4 Numerical Experiments

We have tested the performance of both linear and square penalty SVR methods in two Proben1 regression problems [7] and also in a wind power prediction one. The SVMSeq [13] algorithm was applied in the linear penalty case and the SK algorithm [6] for square penalties. In both cases a Gaussian kernel $k(x, y) = \exp\left(-\|x - y\|^2/2\sigma^2\right)$ was used. The Proben1 problems used were `building`, where hourly electrical energy and hot and cold water consumption in a building are to be predicted, and `flare`, where we want to do the same for the daily number of small, medium and large solar surface flares. Both datasets are given

Table 2. Mean square test errors for `building` and `flare` problems and for wind energy prediction obtained by an MLP and linear and square penalty SVMs. The corresponding best result recorded in the Proben1 database is also given.

Problem	Proben1 best result	MLP	Linear penalty SVM	Square penalty SVM
flare1	0.5283	0.5472	0.5444	0.5431
flare2	0.3214	0.2732	0.2680	0.2662
flare3	0.3568	0.3423	0.3457	0.3552
building1	0.6450	0.4267	0.4369	0.4556
building2	0.2509	0.2696	0.2418	0.2616
building3	0.2475	0.2704	0.2318	0.2525
Wind Power	–	8.33	8.96	8.68

Table 3. Initial sample size, number of Support Vectors of linear and square penalty SVMs and corresponding percentages with respect to sample size for the first dependent variable in the `building` and `flare` problems and for wind energy prediction

Problem	Training set size	Linear penalty SVM	Square penalty SVM
flare1	800	792 (99%)	211 (26.37%)
flare2	800	799 (99.87%)	216 (27.00%)
flare3	800	796 (99.5%)	214 (26.75%)
building1	3156	3156 (100%)	2299 (72.84%)
building2	3156	2988 (94.67%)	1430 (45.31%)
building3	3156	3107 (98.44%)	1243 (39.38%)
Wind Power	1560	583 (37.37%)	315 (20.19%)

in [7] in three variations, numbered 1 to 3, each one with a different arrangement of the training and test sets. On the other hand, we will also work with a real wind power prediction task, where numerical weather predictions from the European Centre for Medium-Range Weather Forecasts (ECMWF, [5]) at time T are used to provide energy production estimates for the Sotavento wind farm [10] located in Galicia (Spain) on 36 hour periods going from the $T+12$ to the $T+48$ hour. The test set was the farm's hourly production in August 2006.

Model performance was measured first by mean square error and the SVM model results were compared with the best ones in the Proben1 database and also with those provided by a single hidden layer multilayer perceptron (MLP). In the wind power problem errors are given as percentages of the farm's installed capacity. As it can be seen from the test results of table 2, all models give similar errors but the square penalty SVM ones are usually slightly larger than those of the linear penalty SVM but comparable to the MLP ones. Notice that, in any case, stopping criteria for linear and square penalty SVMs are different, as SVMSeq performs a gradient ascent over the dual problem while the SK algorithm tries to minimize the function g defined in section 2. On the other hand, the number of support vectors (SVs) obtained using square penalties

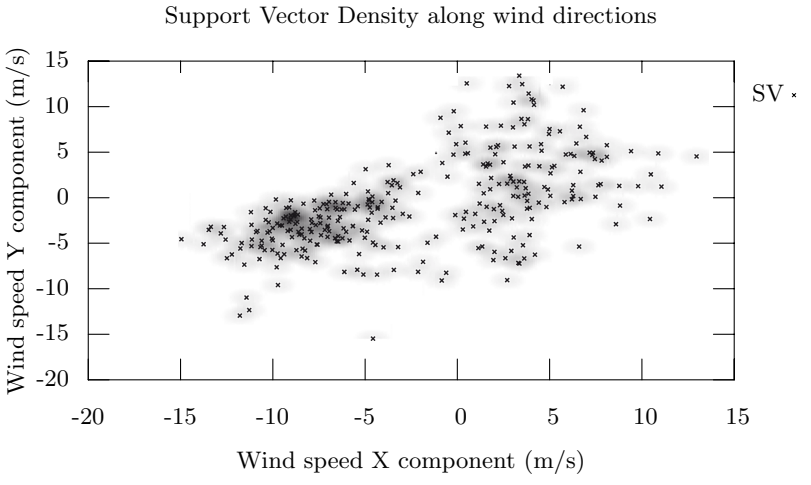


Fig. 1. Placement and density of support vectors plotted along wind speed X and Y components. Darker areas represent a higher support vector density

is in general much smaller than in the case with linear penalties. This is seen in table 3 for the wind energy prediction problem and the first variable to be predicted of each data split for the `building` and `flare` problems (similar results are obtained for the other two dependent variables).

An advantage of SVMs over MLPs for the wind energy prediction problem is the easier interpretation of the resulting model, as the SVs obtained somehow reflect the dominant data and the model's behaviour. For instance, figure 1 depicts the density of support vectors (computed using a Parzen window estimator) along the X and Y wind speed components. On the one hand, the density plot reflects dominant southwest–northeast wind directions in the training database; on the other, it is also clear that model performance over wind speed predictions outside the grey areas is likely to be poor. For instance, no support vectors appear for moderate–to–large wind speeds with northwest and southeast directions; this reflects that wind on these areas has been rare on the training database, but the model will also ignore it in the future.

The effect of the $\tilde{\epsilon}$ tube is depicted for the wind farm training data in figure 2, that shows the distribution of positively (circles) and negatively (crosses) shifted support vectors on a plot with absolute wind speed in the x–axis and energy production (as a percentage of installed power capacity) in the y–axis. As it can be expected, for a given wind speed patterns with negative shifts tend to lie below the positively shifted ones; when this is not the case, it is likely to be due to the presence of outliers. The figure is also somewhat reminiscent of the power curve of wind turbines, which typically have a sigmoidal–like shape with a cut–off for wind speeds above 25 m/s.

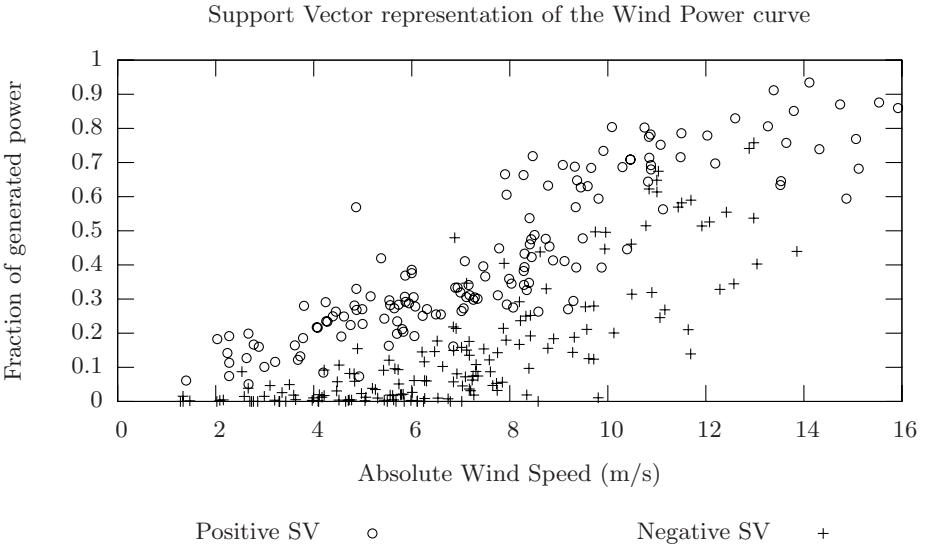


Fig. 2. Positively (circles) and negatively (crosses) shifted SVs over absolute wind (x axis) and percentage of maximum energy output (y axis)

5 Conclusions and Discussion

In this work we have studied support vector regression where tube squared slacks are added as a penalty term to the standard SVM squared weight norm. This has two clear advantages. The first one is that, as it happens with SVM classification, hard and soft SVR can be formally treated in the same way once soft weights and patterns are appropriately extended. The second advantage is that the coefficients of the support vectors obtained have a direct relationship with the width of the tube where these support vectors will fall in. Moreover, and as demonstrated with our numerical experiments, both linear and square penalties seem to give quite similar test errors, while the second models tend to give a smaller number of support vectors (resulting in a faster application on new data).

As pointers to further study, more experimental work is needed for a more precise comparison on linear and square penalty SVR performance. Moreover, the standard SVR formulation has the drawback of having to decide on the extra ϵ parameter on top of the other two usual SVM parameters, namely the penalty factor C and the kernel width σ. For linear penalties, the so–called ν–SVR [8] allows to simultaneously get rid of the C and ϵ parameters by introducing a new parameter ν that, moreover, can be used to control some aspects of the SVM obtained. It may be possible that appropriate square penalty extensions of ν–SVR provide the same benefits. These and similar topics are presently under consideration.

References

1. Bennett, K., Bredensteiner, E.: Geometry in learning. In: Gorini, C., Hart, E., Meyer, W., Phillips T. (eds.). Geometry at Work, Mathematical Association of America, 1997.
2. Bi, J., Bennett, K.: A geometric approach to support vector regression. Neurocomputing 55, 187–220 (2003).
3. Burges, C., Crisp, D.: Uniqueness theorems for kernel methods. Neurocomputing 55, 187–220 (2003).
4. Chang, C., Lin, C.: LIBSVM: a library for support vector machines, http://www.csie.ntu.edu.tw/˜cjlin/LIBSVM.
5. European Centre for Medium-Range Weather Forecasts, http://www.ecmwf.int
6. Franc, V., Hlaváč, V.: An iterative algorithm learning the maximal margin classifier. Pattern Recognition 36, 1985–1996 (2003).
7. Prechelt, L.: Proben1 - A Set of Neural Network Benchmark Problems and Benchmarking Rules, http://digbib.ubka.uni-karlsruhe.de/eva/ira/1994/21.
8. Schölkopf, B., Smola, A., Williamson, R., Bartlett, P.: New support vector algorithms. Neural Computation 12, 1083–1121 (2000).
9. Smola, A., Schölkopf, B.: A tutorial on support vector regression. NeuroCOLT2 Technical Report NC2-TR-1998-030 (1998).
10. Parque Eólico Experimental Sotavento, http://www.sotaventogalicia.com
11. University of California Irvine: UCI-benchmark repository of machine learning data sets, http://www.ics.uci.edu.
12. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Berlin (1995).
13. Vijayakumar, S., Wu, S.: Sequential Support Vector Classifiers and Regression. In: Proc. International Conference on Soft Computing (SOCO'99), pp. 610–619 (1999).

Constructing Accurate Fuzzy Rule-Based Classification Systems Using Apriori Principles and Rule-Weighting

S.M. Fakhrahmad[1], A. Zare[2], and M. Zolghadri Jahromi[2]

[1] Member of Young Researchers Club - Shiraz branch,
Faculty member of Islamic Azad University of Shiraz, Shiraz, Iran
[2] Department of Computer Science & Engineering, Shiraz University, Shiraz, Iran
{mfakhrahmad,aminzare}@cse.shirazu.ac.ir, zjahromi@shirazu.ac.ir

Abstract. A fuzzy rule-based classification system (FRBCS) is one of the most popular approaches used in pattern classification problems. One advantage of a fuzzy rule-based system is its interpretability. However, we're faced with some challenges when generating the rule-base. In high dimensional problems, we can not generate every possible rule with respect to all antecedent combinations. In this paper, by making the use of some data mining concepts, we propose a method for rule generation, which can result in a rule-base containing rules of different lengths. As the next phase, we use rule-weight as a simple mechanism to tune the classifier and propose a new method of rule-weight specification for this purpose. Through computer simulations on some data sets from UCI repository, we show that the proposed scheme achieves better prediction accuracy compared with other fuzzy and non-fuzzy rule-based classification systems proposed in the past.

Keywords: Pattern classification, fuzzy systems, data mining, rule weighting.

1 Introduction

Fuzzy rule-based systems have been widely used on control problems [1,2,3]. One key feature of fuzzy rule-based systems is their comprehensibility because each fuzzy rule is linguistically interpretable. Recently, fuzzy rule-based systems have been applied successfully on classification problems [4,5,6]. The interest in using fuzzy rule-based classification systems (FRBCS) arises from the fact that those systems consider both accuracy and comprehensibility of the classification result at the same time. In fact, error minimization and comprehensibility maximization are two conflicting objectives of these kinds of classification systems and the trade off between these two objectives has been discussed in some recent studies [7,8,9,10].

Basic idea for designing a FRBCS is to automatically generate fuzzy rules from numeric data (i.e., a number of pre-labeled training examples). Hence, rule-base construction for a classification problem always has been a challenging part of it. In this paper, a novel approach for generating a set of candidate rules of each class is presented using data mining principles in which the number of generated rules is

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 547–556, 2007.

reduced dramatically. A compact rule-base is then constructed by selecting a specified number of candidate rules from each class (using a selection metric).

In many studies, antecedent fuzzy sets were generated and tuned by numerical input data for rule-base construction to improve the classification accuracy of FRBCSs. As shown in [11,12], the modification of the membership functions of antecedent fuzzy sets can be replaced by rule weight specification to some extent. Since, the adjustment of membership functions may degrade the interpretability of a FRBCS. In this paper, a learning algorithm is proposed to adjust the weights of the rules (existing in the rule-base) by the training data. This method attends to improve the generalization of FRBCS by minimizing the classification error rate on the training data.

The rest of this paper is organized as follows. In Section 2, a FRBCS is briefly introduced. In Section 3, the process of rule-base construction and the proposed method of generating rules with different lengths are described. Section 4 is devoted to introduction of the proposed method of rule weight learning. In Section 5, the experimental results over artificial and real-life data are shown. Finally, Section 6 concludes the paper.

2 Fuzzy Rule-Based Classification Systems

Various methods have been introduced for fuzzy classification. Let us assume that we have m training patterns $x_p = (x_{p1},\ldots,x_{pn})$, $p = 1,2,\ldots,m$ from M different classes where x_p is an n-dimensional vector of attributes in which x_{pi} is the i-th attribute value of the p-th training pattern ($i = 1,2,\ldots,n$). For our M-class, n-dimensional classification problem, we use fuzzy if-then rules of the form below:

$$\text{Rule } R_q\text{: If } x_1 \text{ is } A_{q1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{qn} \text{ then class } C_q \text{ with } CF_q \ , \tag{1}$$

where R_q is the label of the q-th fuzzy if-then rule, $\mathbf{x} = (x_1,\ldots,x_n)$ is n-dimensional vector of a pattern, A_{qi} presents an antecedent fuzzy set, C_q is a class label, and CF_q is the weight assigned to the q-th rule. In [13], fuzzy rules of other types are introduced. To calculate the compatibility grade of each training pattern $\mathbf{x}_p$ with the antecedent part of the rule $\mathbf{A}_q = (A_{q1},\ldots,A_{qn})$, we use the product operator as follows:

$$\mu_{A_q}(\boldsymbol{x}_p) = \mu_{A_{q1}}(x_{p1}) \cdot \mu_{A_{q2}}(x_{p2}) \cdot \ldots \cdot \mu_{A_{qn}}(x_{pn}), p = 1,2,\ldots,m \ , \tag{2}$$

where $\mu_{Aqi}(x_{pi})$ is the compatibility grade of x_{pi} with fuzzy membership function A_{qi}. To determine the consequent class of the q-th rule C_q, we measure the confidence degree of the association rule "$\mathbf{A}_q \Rightarrow$ Class h" from the field of data mining for each class, where $\mathbf{A}_q$ is a multi-dimensional fuzzy set representing the antecedent conditions and h is a class label. Confidence of a fuzzy association rule R_q is defined as follows:

$$c(A_q \Rightarrow \text{Class h}) = \sum_{X_p \in \text{Class h}} \mu_{A_q}(\boldsymbol{x}_p) \Big/ \sum_{p=1}^{m} \mu_{A_q}(\boldsymbol{x}_p), h = 1,2,\ldots,M \ , \tag{3}$$

where $\mu_{Aq}(X_p)$ is the compatibility grade of pattern X_p with the antecedent part of the rule R_q, m is the number of training patterns and C_q is a class label. The class with maximum confidence degree is identified to determine the consequent class C_q:

$$q = \text{argmax}\{c(A_q \Rightarrow \text{Class h}) \mid h = 1, 2, \ldots, M\} \tag{4}$$

An input pattern is classified regarding to the consequent class of the winner rule. By using rules of the form (1), a weight assigned to each rule is used to find the winner rule. Rule weighting has a profound effect on the classification ability of FRBCSs. In [14], several methods of rule weighting have been introduced. In this paper, we use a learning mechanism to find the weight of each rule.

3 Rule-Base Construction

For an M-class problem in an n-dimensional feature space, assume that m labeled patterns $X_p=[x_{p1}, x_{p2}, \ldots, x_{pn}]$, $p=1, 2, \ldots, m$ from M classes are given. A simple approach for generating fuzzy rules is to partition the domain interval of each input attribute using a pre-specified number of fuzzy sets (i.e., grid partitioning). Some examples of this partitioning (using triangular membership functions) are shown in Fig. 1.

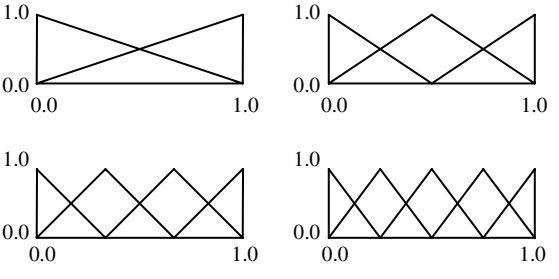


Fig. 1. Different partitioning of each feature axis

Given a partitioning of pattern space, one approach is to consider every possible combination of antecedents to generate the fuzzy rules. The problem with grid partitioning is that an appropriate partitioning of each attribute is not usually known. One solution is to simultaneously consider different partitions, as shown in Fig. 1. That is, for each attribute, one of the 14 fuzzy sets shown in Fig. 1 can be used when generating a fuzzy rule. The problem is that for an n-dimensional problem, 14^n antecedent combinations have to be considered. It is impractical to consider such a huge number of antecedent combinations when dealing with high dimensional problems.

One solution for the above problem has already been presented by adding the fuzzy set "don't care" to each attribute. The membership function of this fuzzy set is defined as $\mu_{don't\ care}(x) = 1$ for all values of x. The trick is not to consider all antecedent combinations (which is now 15^n) and only short fuzzy rules having a limited number

of antecedent conditions are generated as candidate rules. For example, fuzzy rules having only two or less antecedent fuzzy sets (excluding don't care) are investigated.

It seems that ignoring majority of possible antecedent combinations for rule generation would degrade the accuracy of the FRBCS. On the other hand, increasing the limitation of two or less antecedent fuzzy sets may be impossible in some cases. As an instance, for a high-dimensional data set such as Sonar (available in UCI-ML repository) which is a 2-class data set, containing 60 attributes, number of all possible fuzzy rules of length 4, considering 15 fuzzy membership functions for each attribute is $(60 \times 15)^4 \times 2$ which is more than 10^{12}. Since measuring confidence and support values for each rule involves scanning all training data patterns, construction of a rule-base containing such rules seems to be very difficult or even impossible. That's why the existing classifiers are able to generate rules having at most three antecedent conditions (excluding don't care) in such cases. This limitation prevents from some useful rules, which would have positive effects on the classification accuracy, to be present in the rule-base. As mentioned in the last paragraph, by increasing the number of antecedents, the rule set grows exponentially. Moreover, within a very large number of rules, usually a small fraction of rules are acceptable. Thus, in many cases, a considerable time is devoted to useless computations.

The purpose of the solution presented in this paper, is to avoid the exponential growth of the rule sets in each step. In this approach, we do not generate rules that are hardly probable to be interesting. The method is based over two data mining principles, used for mining frequent item sets:

1) Increasing the length of an item set, the support value will not improve.
2) A set of n items is probable to be frequent (have a good support), if and only if all of its subsets of size n-1 are frequent (the Apriori principle).

The common usage of the above principles is in data warehouses, to find itemsets with good supports (i.e., set of items that have frequently occurred together). In this work, we observe them from a different viewpoint and use them to find fuzzy rules having good supports.

3.1 Generating Rules with 1 or 2 Antecedents

As mentioned before, A major purpose in this paper is to propose a solution that enables us to generate fuzzy rules with any number of antecedents, i.e., There would be no restriction on the number of antecedents especially for high dimensional data sets (the problem which originates from the exponential growth of rule-base by increasing the number of features). For this purpose, we consider the well-known evaluation measure, *Support* as the primary factor for rule filtering. In equation (5), a simple definition for the fuzzy aspect of the *Support* measure is presented.

$$s(A_j \Rightarrow \text{Class } h) = \frac{1}{m} \sum_{X_p \in \text{Class h}} \mu_{A_j}(X_p), \tag{5}$$

where $\mu_j(X_p)$ is the compatibility degree of X_p with the antecedent part of the rule R_j, m is the number of training patterns and h is a class label. After determining a minimum support threshold (denoted by *MinSupp*), a set of 1-dimensional rules

(containing one antecedent), is generated. This set is then filtered by selecting only rules having a support value above the *MinSupp*. Combining the rules within this set in the next step, results in the set of 2-dimensional candidate rules. The reason of this issue (that we just combine rules having good supports) refers to the first principle mentioned in Section 2. In other words, the 1-dimensional rules which are pruned through the first step because of their bad supports, can not absolutely lead to 2-dimensional rules with good supports and thus there is no need to consider them. Another key point in combination of a pair of 1-dimensional rules is the conditions under which the rules can be combined:

1) The rules must not contain similar antecedents on their left-hand sides.
2) The consequent classes of the two rules must be identical. Similarly, the resulting rule set is filtered with respect to the *MinSupp* value. However, note that the rules being selected according to their higher support values are just candidate rules and may be rejected in the next step. The rule selection metric will be discussed later.

3.2 Generating Rules of Higher Dimensions

In order to generate rules containing more than two antecedents, a similar procedure is followed. However, in this case, both of the principles (used for mining frequent item sets) must be regarded. Generating 3-dimensional rules is accomplished using the 1 and 2-dimensional candidate rules. Any possible combination of the rules from these two sets, having the same consequent and not containing common antecedents would be a 3-dimensional candidate rule. The second principle is used to avoid the time-consuming evaluation of some useless rules (which can not have high support values). A rule resulting from a combination will be evaluated only if all of its 2-dimensional sub-rules[1] are present in the candidate set of the previous stage (i.e., all the sub-rules have good supports). Otherwise, we do not measure the support of the rule, since it can not be even a candidate rule. As an example, the support of the rule R: If X1 is A1 and X2 is A2 and X3 is A3 → C1 is computed only if all the following sub-rules have good supports:

If X1 is A1 and X2 is A2 → C1
If X1 is A1 and X3 is A3 → C1
If X2 is A2 and X3 is A3 → C1

Similarly, generating an n-dimensional rule is performed by the combination of n-1 and 1-dimensional candidate rules.

An important challenge here is to find a solution for this problem: How should we control the presence of all sub-rules in order to avoid efficiency reduction. Moreover, combining 1 and n-dimensional rules (even with respect to the mentioned conditions) may lead to some repeating combinations. Another challenging problem is how to avoid generating repeating rules, which could be so helpful to the efficiency of the process. To achieve these two goals, we make use of the efficiency of SQL and accomplish the primary phases of the rule generation process using this language.

[1] Rule R1 is called to be a sub-rule of R2 iff R1 has less antecedents than R2 and every antecedent of R1 is present in R2, too.

Following the above process, it will also be possible to generate rules having 4 and more antecedents, for any data set having arbitrary number of features.

Although the set of rules is pruned to some extent, in some cases the number of rules is still large. This problem gets more sensible as we increase the number of antecedents. In order to obtain a more concise data set, we divide the set of candidate rules into *M* distinct groups, according to their consequents (*M* is the number of classes). The rules in each group are sorted by an appropriate evaluation factor and the final rule-base is constructed by selecting *p* rules from each class, i.e., in total, *M.p* rules are selected. Many evaluation measures have already been proposed. In this work, we use the measure proposed in [15] as the rule selection metric, which evaluates the rule $A_j \Rightarrow class\ C_j$ through the following equation:

$$e(R_j) = \sum_{X_p \in \text{Class } C_j} \mu_{A_q}(\boldsymbol{x}_p) - \sum_{X_p \notin \text{Class } C_j} \mu_{A_q}(\boldsymbol{x}_p) \tag{7}$$

4 The Proposed Method for Rule Weighting

Initially, all rules are assumed to have a weight of one (i.e. CF_k=1, K=1,2,...,N). In this section we propose an algorithm to assign some real numbers (in the interval [1,∞)) as the rule weights using the training patterns. The rule-weighting process for a typical rule, R_i, can be organized into the following steps:

1. Specify the center of the rule's covering area, sort the training patterns within this area in ascending order of their distance from the center (The first pattern in the sorted list is the most compatible one with the rule).
2. Scan the patterns through the sorted list until a pattern (X_n) from the negative class (any class except the rule's target class) is met (The enemy pattern with maximum compatibility degree is found).
3. Call the pattern just before the enemy pattern in the list X^* and Find its compatibility with the rule R_i ($\mu_i(X^*)$).
4. Compute the rule's weight (CF_i) using the following equation:

$$CF_i = 1/\mu_i(X^*) \tag{8}$$

This algorithm obtains a real number in the interval [1,∞) as the weight of each rule. However, in this issue, two exceptional cases may occur:

1. The first pattern in the sorted list is an enemy pattern. For this case, we set the value of 1 to $\mu_i(X^*)$ and thus the rule's weight will not change from 1.
2. There is no enemy pattern in the covering area of the rule (i.e., an interesting case). For this case, we chose the compatibility degree of the last pattern in the sorted list for $\mu_i(X^*)$, i.e., $\mu_i(X^*) = \mu_i$(last pattern). Since the last pattern has the minimum compatibility with the rule, a higher weight is given to such rules.

In this method, no rule is given a weight of 0. Thus, the number of rules does not change through this weighting process. As the number of partitions of each feature

increases, the performance of the system approaches the performance of weighted K-NN method, while having the extra advantage of interpretability, especially for low-dimensional data sets.

4.1 Finding the Optimal Decision Boundaries

The method proposed in Section 3, increases the classification accuracy of each fuzzy rule over training data up to 100%. This is accomplished by tuning the boundaries for the decision area of each rule through assigning a weight to it. It can be predicted that the generalization ability of the classifier will be improved, too. However, there is really no reason that we will get the optimal results for generalization accuracy, through this issue. The main reason refers to some probable noisy or exceptional patterns or in case of data sets with highly ovelapped classes. Our proposed method can easily become more flexible (for noisy data) by making a small change to it. After determining a threshold for the rule accuracy, we do not stop the scanning of the sorted list when meeting the first enemy pattern. Instead, we extend the decision boundary of the rule until we reach an enemy pattern that makes the rule's accuracy become less than the specified threshold. In other words we let a few number of enemy patterns to exist in the decision area of each rule. This can be more effective for noisy-nature data sets.

To find the best accuracy threshold for a typical data set, different threshold values can be tested through the discussed method and the optimal value will be obtained by comparing the generalization accuracies of different cases.

5 Experimental Results

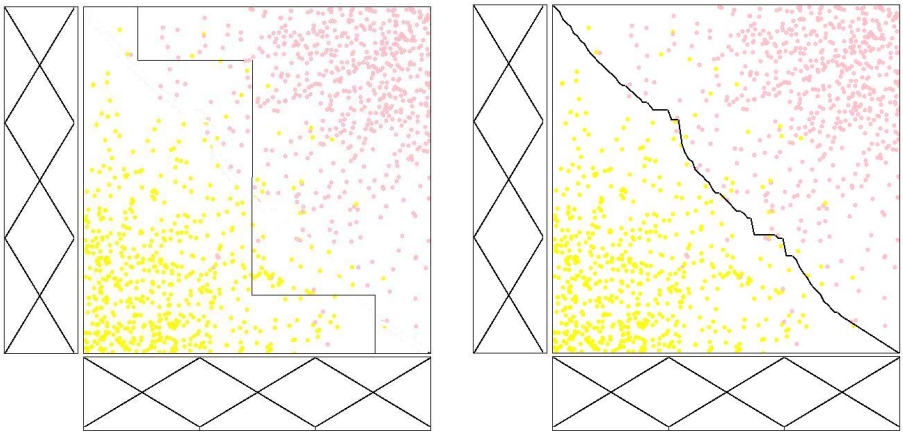
In order to evaluate the performance of the proposed scheme, we arranged two sets of experiment**,** over artificial and real-life data sets. The results were compared with the non-weighted rule-base classifier in each case.

5.1 Artificial Data

In this experiment, we generated 1200 data points of two classes, namely 1 and 2. The data of the first class had a normal distribution, with $\mu = (0,0)$ and $\sigma = 0.3$, whereas the data of the second class had a normal distribution, with $\mu = (1,1)$ and $\sigma = 0.3$. Using 4 triangular fuzzy sets, we first run the non-weighted and then the weighted rule-base classifier over this data set and found the discriminating line in each case. It is known that the closer the discriminating line to the diagonal, the higher generalization ability of the classifier. The results of this experiment are shown in Fig. 2.

5.2 Real-Life Data Sets

In order to assess the performance of the proposed scheme over some real-life data, we used the data sets shown in Table 1 available from UCI ML repository. To



a) Non-weighted rule-base classifier b) Weighted rule-base classifier

Fig. 2. Visualized effect of the proposed rule-weighting technique compared to the non-weighted case for a 2-class, Normal-distributed data set

construct an initial rule-base for a specific data set, a number of equi-length rules (number of antecedents equaling to the number of features), having at least one training pattern in their decision areas were generated. In order to assess the effect of the proposed scheme in comparison with its alternatives, we used 10CV technique which is a case of n-fold cross validation.

In the first part of the experiment, the generalization accuracy of the initial rule-base (before rule weighting) was measured.

In the second part, our rule-weighting method was evaluated without considering any accuracy threshold, as discussed in Section 4. In other words, the threshold was set to 1. The results, shown in Table 2 narrate from a positive effect for the weighting method over the generalization ability.

Finally, in the third part, for each data set, we tried different values of the accuracy threshold by changing it from 1 down to 0.4 (by the step size of 0.05). Using the LVO (Leave One Out) technique, in each case, the error rate was measured using training data. The best threshold (leading to the best result) was then selected to evaluate the generalization ability over that data set using the 10CV technique.

The error rates of the classifier using the optimal value of θ for each data set are presented in Table 2. In this table, our proposed method is compared with another successful rule-based method as benchmark results called C4.5 reported by Elomaa and Rousu [7]. It is also compared to four different weighting methods defined by Ishibuchi in [12]. In all cases, the rule selection is performed using the single winner method. As shown in Table 2, except in one case, the proposed classifier in this paper results in better classification rates, compared to the best results already achieved by C4.5. As seen, the proposed method achieves better results in comparison with Ishibuchi metrics in all experimental cases.

Table 1. Some statistics of the data sets used in our computer simulations

Data set	Number of attributes	Number of patterns	Number of classes
Iris	4	150	3
Wine	13	178	3
Thyroid	5	215	3
Sonar	60	208	2
Bupa	6	345	2
Pima	8	768	2
Glass	9	214	6

Table 2. Classification Error rates of the proposed classifier using the optimal threshold values in comparison with threshold of 1, the non-weighted rule-base classifier, Ishibuchi weighting methods and the C4.5 method for data sets of Table 1

Data sets	Error Rates (%)							
	No Weight	proposed method		Ishibuchi Metrics				C4.5 (best results)
		$\theta = 1$	Optimal θ	Metric1	Metric2	Metric3	Metric4	
Iris	5	3.6	**3.6**	4.2	4.6	4.4	4.2	5.1
Wine	6.1	5.2	**5.1**	8.4	8.4	5.6	6.7	5.6
Pima	27.8	25.7	**24.3**	26.3	26.5	27.8	26.5	25
Bupa	39	37.8	**37.8**	39	38.4	38.8	38.8	38.2
Thyroid	8.5	4.1	**4.1**	6.7	5.8	6.3	6.1	6.7
Glass	35	34.9	**33.3**	41.1	40.2	40.6	44.4	27.3
Sonar	11.2	5	**4.1**	11	10.8	10.6	10.8	23.3

6 Conclusions

In this paper, a novel method of rule-base construction using data mining principles and a rule weighting mechanism was proposed. Using the proposed method for rule generation, it will be possible to generate rules having different lengths, efficiently. It is much more useful when dealing with high dimensional data sets, were the existing methods are not able to generate rules containing more than 2 or 3 antecedent conditions. We also proposed a new method of rule-weight specification in order to tune the classifier. As the number of partitions of each feature increases, the generalization ability of the system competes and even precedes the weighted K-NN method. Moreover, the proposed scheme is a FRBCS and has the advantage of interpretability, especially for low-dimensional data sets.

We also proposed a mechanism to find the optimal rule weights, which is much more useful in case of noisy or highly overlapped data sets, in order to prevent from overfitting of the learned classifier.

We used seven data sets from UCI-ML repository to assess the performance of the learning scheme. Simulation results on thsese data sets showed that the method can be used to construct a rule-base with a good generalization ability. The effect of rule weights could be seen, clearly, through this experiment. We also showed that the proposed method is more effective in reducing the error rate of the classifier in

comparison with all weighting metrics introduced by Ishibuchi and also comparing to the C4.5 as a successful rule-based method.

References

1. Arima, M., Hara, E.H., Katzberg, J.D.: A fuzzy logic and rough sets controller for HVAC systems. In: The IEEE WESCANEX, pp. 133–138. New York (1995)
2. Cordón, O., Herrera, F., Peregrín, A.: Applicability of the fuzzy operators in the design of fuzzy logic controllers. Fuzzy Sets and Systems 86(1), 15–41 (1997)
3. Glorennec, P.Y.: Application of fuzzy control for building energy management. Building Simulation. International Building Performance Simulation Association, France 1, 197–201 (1991)
4. Bárdossy, A., Duckstein, L.: Fuzzy rule-based modeling with applications to geophysical. biological and engineering systems. CRC Press, Boca Raton (1995)
5. Bezdek, J.C., Pal, S.K.: Fuzzy Models for Pattern Recognition. In: Methods that Search for Structures in Data, IEEE Press, Boca Raton (1992)
6. Chi, Z., Yan, H., Pham, T.: Fuzzy algorithms with applications to image processing and pattern recognition. World Scientific, New York (1996)
7. Jin, Y.: Fuzzy Modeling of High-dimensional Systems. Complexity Reduction and Interpretability Improvement. IEEE Trans. on Fuzzy Systems, 212–221 (2000)
8. Jin, Y., Von Seelen, W., Sendhoff, B.: On Generating FC3 Fuzzy Rule Systems from Data Using Evolution Strategies. IEEE Trans. on Systems, Man and Cybernetics - Part B: Cybernetics 29, 829–845 (1999)
9. Roubos, H., Setnes, M.: Compact and Transparent Fuzzy Models and Classifiers Through Iterative Complexity Reduction. IEEE Trans. on Fuzzy Systems, 516–524 (2001)
10. Setnes, M., Babuska, R., Verbruggen, B.: Rule-based Modeling: Precision and Transparency. IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews 28, 165–169 (1998)
11. Nauck, D., Kruse, R.: How the learning of rule weights affects the interpretability of fuzzy systems. In: 7th IEEE International Conference on Fuzzy Systems, pp. 1235–1240. Anchorage (1998)
12. Ishibuchi, H., Yamamoto, T.: Rule weight specification in fuzzy rule-based classification systems. IEEE Trans. on Fuzzy Systems 13(4), 428–435 (2005)
13. Ishibuchi, H., Nozaki, K., Tanaka, H.: Distributed representation of fuzzy rules and its application to pattern classification. Fuzzy Sets and Systems 52, 21–32 (1992)
14. Ishibuchi, H., Nakashima, T., Morisawa, T.: Voting in Fuzzy Rule-Based Systems for Pattern Classification problems. Fuzzy Sets and Systems 103, 223–238 (1999)
15. Ishibuchi, H., Yamamoto, T.: Comparison of heuristic criteria for fuzzy rule selection in classification problems. Fuzzy Optimization and Decision Making 3(2), 119–139 (2004)

Visualization of Topology Representing Networks

Agnes Vathy-Fogarassy[1], Agnes Werner-Stark[1], Balazs Gal[1], and Janos Abonyi[2]

[1] University of Pannonia, Department of Mathematics and Computing, P.O. Box 158, Veszprem, H-8201 Hungary
vathya@almos.uni-pannon.hu
[2] University of Pannonia, Department of Process Engineering, P.O. Box 158, Veszprem, H-8201 Hungary
abonyij@fmt.uni-pannon.hu,
http://www.fmt.uni-pannon.hu/softcomp

Abstract. As data analysis tasks often have to face the analysis of huge and complex data sets there is a need for new algorithms that combine vector quantization and mapping methods to visualize the hidden data structure in a low-dimensional vector space. In this paper a new class of algorithms is defined. Topology representing networks are applied to quantify and disclose the data structure and different nonlinear mapping algorithms for the low-dimensional visualization are applied for the mapping of the quantized data. To evaluate the main properties of the resulted topology representing network based mapping methods a detailed analysis based on the wine benchmark example is given.

1 Introduction

In the majority of practical data mining problems high-dimensional data has to be analyzed. Because humans simply can not see high-dimensional data, it is very informative to map and visualize the hidden structure of complex data set in a low-dimensional space. The goal of dimensionality reduction is to map a set of observations from a high-dimensional space (D) into a low-dimensional space (d, $d \ll D$) preserving as much of the intrinsic structure of the data as possible. Three types of dimensionality reduction methods can be distinguished: (i) metric methods try to preserve the distances of the data defined by a metric, (ii) non-metric methods try to preserve the global ordering relations of the data, (iii) other methods that differ from the previously introduced two groups. Principal Component Analysis (PCA) [6,7], Sammon mapping (SM) [14] and multidimensional scaling (MDS) [2] are widely used dimensionality reduction methods. Sammon mapping minimizes the Sammon stress function by the gradient descent method, while the classical MDS though similarly minimizes the cost function, but it uses an eigenvalue decomposition based (single step) algorithm. So e.g. the optimization algorithm used by the Sammon mapping can stuck in local minima,hence it is sensitive to the initialization. The MDS has

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 557–566, 2007.

a metric and a non-metric variant, thereby it can also preserve the pairwise distances or the rank ordering among the data objects.

In the literature there are several neural networks proposed to visualize high-dimensional data in low-dimensional space. The Self-Organizing Map (SOM) [8] is one of the most popular artificial neural networks. The main disadvantage of SOM is that it maps the data objects into a topological ordered grid, thereby it is needed to utilize complementary methods (coloring scheme such as U-matrix) to visualize the relative distances between data points on the map. The Visualization Induced Self-Organizing Map (ViSOM) [18] is an effective extension of SOM. ViSOM is an unsupervised learning algorithm, which is proposed to directly preserve the local distance information on the map. ViSOM preserves the inter-point distances as well as the topology of data, therefore it provides a direct visualization of the structure and distribution of the data. ViSOM constrains the lateral contraction forces between neurons and hence regularizes the interneuron distances so that distances between neurons in the data space are in proportion to those in the input space [18]. The motivation of the development of the ViSOM algorithm was similar to the motivation of this work, but here the improvement of the Topology Representing Network based data visualization techniques are in focus.

Dimensionality reduction methods in many cases are confronted with low-dimensional structures nonlinearly embedded in the high-dimensional space. In these cases the Euclidean distance is not suitable to compute distances among the data points. The geodesic distance [1] is more suitable to catch the pairwise distances of objects lying on a manifold, because it is computed in such a way that it always goes along the manifold. To compute the geodesic distances a graph should be built on the data. The geodesic distance of two objects is the sum of the length of the edges that lie on the shortest path connecting them. Although most of the algorithms utilize the neighborhood graphs for the construction of the representative graph of the data set, there are other possibilities to disclose the topology of the data, too. Topology representing networks refers to a group of methods that generate compact, optimal topology preserving maps for different data sets. Topology representative methods combine the neural gas (NG) [11] vector quantization method and the competitive Hebbian learning rule [5].

There are many methods published in the literature proposing to capture the topology of the given data set. Martinetz and Shulten [12] showed how the simple competitive Hebbian rule forms Topology Representing Network (TRN). Dynamic Topology Representing Networks (DTRN) were introduced by Si at al. [15]. In their method the topology graph incrementally grows by adding and removing edges and vertices. Weighted Incremental Neural Network (WINN) [13] produces a weighted connected net. This net consists of weighted nodes connected by weighted edges. Although, the TRN, DTRN and WINN algorithms are quite similar, the TRN algorithm gives the most robust representation of the data.

The aim of this paper is to analyze the different topology representing network based data visualization techniques. For this purpose we round up the techniques being based on this method and perform an analysis on them. The analysis

compares the mapping qualities in the local environment of the objects and the global mapping properties.

The organization of this paper is as follows. Section 2 gives an overview of the Topology Representing Network and introduces the related mapping methods. Section 3 introduces the measurement of the mapping quality and gives application example to show the results of the analysis. Section 4 concludes the paper.

2 Topology Representing Network Based Mapping Algorithms

2.1 Topology Representing Network

Given a set of data ($\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \mathbb{R}^D$, $i = 1, \ldots, N$) and a set of codebook vectors ($\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n\}$, $\mathbf{w}_i \in \mathbb{R}^D$, $i = 1, \ldots, n$) ($N > n$). TRN algorithm distributes the pointers $\mathbf{w}_i$ between the data objects by neural gas algorithm, and forms connections between them by applying competitive Hebbian rule. The algorithm of the Topology Representing Network firstly selects some random points (units) in the input space. The number of units (n) is a predefined parameter. The algorithm then iteratively selects an object from the input data set randomly and moves all units closer to this pattern. After this step, the two units closest to the randomly selected input pattern will be connected. Finally, edges exceeding a predefined age are removed. This iteration process is continued until a termination criterion is satisfied. The run of the algorithm results in a Topology Representing Network that means a graph $G = (W, C)$, where W denotes the nodes (codebook vectors, neural units, representatives) and C yields the set of edges between them. The detailed description of the TRN algorithm can be found in [12].

The algorithm has many parameters. The number of the iterations (t_{max}) and the number of the codebook vectors (n) are determined by the user. Parameter λ, step size ε and lifetime T are dependent on the number of the iterations. This time dependence can be expressed by the following general form:

$$g(t) = g_i \left(\frac{g_f}{g_i} \right)^{t/t_{max}} \tag{1}$$

where g_i denotes the initial value of the variable, g_f denotes the final value of the variable, t denotes the iteration counter, and t_{max} denotes the maximum number of iterations. (For example for parameter λ it means: $\lambda(t) = \lambda_i(\lambda_f/\lambda_i)^{t/t_{max}}$.) Paper [12] gives good suggestions to tune these parameters.

In the literature few methods are only published, that utilize the topology representing networks to visualize the data set in the low-dimensional vector space. The Online Visualization Neural Gas (OVI-NG) [3] is a nonlinear projection method, in which the codebook positions are adjusted in a continuous output space by using an adaptation rule that minimizes a cost function that favors the local distance preservation. As OVI-NG utilizes Euclidean distances to map the

data set it is not able to disclose the nonlinearly embedded data structures. The Geodesic Nonlinear Projection Neural Gas (GNLP-NG) [4] algorithm is an extension of OVI-NG, that uses geodesic distances instead of the Euclidean ones. Abreast with these algorithms J.Abonyi and A. Vathy-Fogarassy from among the authors of this article have developed a new group of the mapping methods, called Topology Representing Network Map (TRNMap) [16]. TRNMap also utilizes the Topology Representing Network and the resulted graph is mapped by MDS into a low-dimensional vector space. Hence TRNMap utilizes geodesic distances during the mapping process, it is a nonlinear mapping method, which focuses on the global structure of data. As the OVI-NG is not able to disclose the nonlinearly embedded manifolds in the following we will not deal with this method.

2.2 Geodesic Nonlinear Projection Neural Gas

The GNLP-NG algorithm is a nonlinear mapping procedure, which includes the following two major steps: (1) creating a topology representing network to depict the structure of the data set, and (2) mapping this approximate structure into a low-dimensional vector space. The first step utilizes the neural gas vector quantization method to define the codebook vectors ($\mathbf{w}_i$) in the input space, and it uses the competitive Hebbian rule for building a connectivity graph linking these codebook vectors. The applied combination of the neural gas method and the Hebbian rule differs slightly from the TRN algorithm: it connects not only the first and the second closest codebook vectors to the randomly selected input pattern, but it creates connection between the k-th and the $k+1$-th nearest units ($1 \leq k \leq K$), if it does not exist already, and the $k+1$-th nearest unit is closer to the k-th nearest unit than to the unit closest to the randomly selected input pattern. The parameter K is an accessory parameter compared to the TRN algorithm, and in [4] it is suggested to set to $K = 2$. Furthermore GNLP-NG increments not only the ages of all connections of the nearest unit, but it also extends this step to the k-th nearest units.

During the mapping process the GNLP-NG algorithm applies an adaptation rule for determining the positions of the codebook vectors ($\mathbf{w}_i$, $i = 1, 2, \ldots, n$) in the (low-dimensional) projection space. The mapped codebook vectors are called codebook positions ($\mathbf{z}_i$, $i = 1, 2, \ldots, n$). The mapping process can be summarized as follows:

1. Compute the geodesic distances between the codebook vectors based on the connections of the previously calculated topology representing network. Set $t = 0$.
2. Initialize the codebook positions $\mathbf{z}_j$, randomly.
3. Select an input pattern $\mathbf{x}$ with equal probability for each $\mathbf{x}$. Increase the iteration step $t = t + 1$.
4. Find the codebook vector $\mathbf{w}_{i_0}$ in input space that is closest to $\mathbf{x}$.
5. Generate the ranking ($m_j \in 0, 1, \ldots, n-1$) for each codebook vector $\mathbf{w}_i$ with respect to the $\mathbf{w}_{i_0}$.

6. Update the codebook positions in the output space:

$$\mathbf{z}_i^{new} = \mathbf{z}_i^{old} + \alpha e^{-\left(\frac{m_j}{\sigma(t)}\right)^2} \frac{(D_{i_0,i} - \delta_{i_0,i})}{D_{i_0,i}} (\mathbf{z}_{i_0} - \mathbf{z}_i) \tag{2}$$

7. If $t < t_{max}$ go back to step 3.

Parameter α is the learning rate, σ is the width of the neighborhood, and they typically decrease with the number of iterations t, in the same way as Equation 1. $D_{j,k}$ denotes the Euclidean distance of the codebook vectors $\mathbf{z}_j$ and $\mathbf{z}_k$ defined in the output space, $\delta_{j,k}$ yields the geodesic distance between codebook vectors $\mathbf{w}_j$ and $\mathbf{w}_k$ measured in the input space, and m_j yields the ranking value of the codebook vector $\mathbf{w}_j$. Paper [4] gives an extension to the GNLP-NG, to tear or cut the graphs with non-contractible cycles.

2.3 Topology Representing Network Map

Topology Representing Network Map (TRNMap) refers to a group of nonlinear mapping methods, which combines the TRN algorithm and the MDS method to visualize the data structure to be analyzed. The algorithm has the following major steps: (0) data normalization to avoid the influence of the range of the attributes, (1) creating the Topology Representing Network of the input data set, (2) if the resulting graph is unconnected, the algorithm connects the subgraphs together, (3) calculation of the pairwise graph distances, (4) mapping the modified TRN, (5) creating the component planes. A component plane displays the value of one component of each node. If the input data set has D attributes, the TRNMap component plane includes D different maps according to the D components. The structure of these maps is the same as the TRNMap map, but the nodes are represented in greyscale. The mapping process of the TRNMap algorithm can be carried out by the use of metric or non-metric multidimensional scaling, as well.

The Topology Representing Network Map Algorithm

0. Normalize the input data set $\mathbf{X}$.
1. Create the Topology Representing Network of $\mathbf{X}$ by the use of the TRN algorithm. Yield $M^{(D)} = (W, C)$ the resulting graph, let $\mathbf{w}_i \in W$ the representatives (nodes) of $M^{(D)}$. If exists an edge between the representatives $\mathbf{w}_i$ and $\mathbf{w}_j$ ($\mathbf{w}_i, \mathbf{w}_j \in W$, $i \neq j$), $c_{i,j} = 1$, otherwise $c_{i,j} = 0$.
2. If $M^{(D)}$ is not connected, connect the subgraphs in the following way:
   While there are unconnected subgraphs ($m_i^{(D)} \subset M^{(D)}$, $i = 1, 2, \ldots$):
   (a) Choose a subgraph $m_i^{(D)}$.
   (b) Let the terminal node $\mathbf{t}_1 \in m_i^{(D)}$ and its closest neighbor $\mathbf{t}_2 \notin m_i^{(D)}$ from:
   $\|\mathbf{t}_1 - \mathbf{t}_2\| = min\|\mathbf{w}_j - \mathbf{w}_k\|, \quad \mathbf{t}_1, \mathbf{w}_j \in m_i^{(D)}, \quad \mathbf{t}_2, \mathbf{w}_k \notin m_i^{(D)}$
   (c) Set $c_{\mathbf{t}_1,\mathbf{t}_2}$=1.
   Yield $M^{*(D)}$ the modified $M^{(D)}$.

3. Calculate the geodesic distances between all $\mathbf{w}_i, \mathbf{w}_j \in M^{*(D)}$.
4. Map the graph $M^{(D)}$ into a 2-dimensional vector space by MDS based on the graph distances of $M^{*(D)}$.
5. Create component planes for the resulting Topology Representing Network Map based on the values of $\mathbf{w}_i \in M^{(D)}$.

The parameters of the TRNMap algorithm are the same as those of the Topology Representing Networks algorithm. The TRNMap algorithm has different variations based on the mapping used. If the applied MDS is a metric MDS method, the mapping process will preserve the pairwise distances of the objects. On the other hand, if the TRNMap algorithm applies a non-metric MDS, the resulted map tries to preserve the global ordering relations of the data.

Table 1 gives a systematic overview of GNLP-NG, metric TRNMap (DP_TRNMap, DP from distance preserving) and non-metric TRNMap (NP_TRNMap, NP from neighborhood preserving). It also includes the combination of the non-metric TRNMap and the GNLP-NG algorithms (NP_TRNMap-GNLP_NG), which means the fine tuning of the non-metric TRNMap with the GNLP-NG as follows: after the running of the non-metric TRNMap the projected codebook vectors were 'fine tuned' by the mapping of the GNLP-NG algorithm. This table comparable summarizes the applied topology learning methods, distance measures, and mapping techniques.

Table 1. Systematic overview of the Topology Representing Network based mapping methods

Algorithm	topology learning	distance measure	mapping
GNLP_NG	modified TRN	geodesic	iterative adaptation rule
DP_TRNMap	TRN	geodesic	metric MDS
NP_TRNMap	TRN	geodesic	non-metric MDS
NP_TRNMap-GNLP_NG	TRN	geodesic	combined non-metric MDS and iterative adaptation rule

3 Analysis of the Topology Representing Network Based Mapping Methods

The aim of this section is to analyze the Topology Representing Network based mapping methods that are able to unfold the nonlinearly embedded manifolds. We have shown that GNLP-NG and TRNMap algorithms are nonlinear mapping methods, which utilize a topology representing network to visualize the high-dimensional data structure in the low-dimensional vector space. Because the GNLP-NG method utilizes a non-metric mapping procedure, and the TRNMap also has a non-metric variant, their mapping qualities of the neighborhood preservation can be compared.

3.1 Mapping Quality

A projection is said to be *trustworthy* [9,17] if the nearest neighbors of a point in the reduced space are also close in the original vector space. Let n be the number of the objects to be mapped, $U_k(i)$ be the set of points that are in the k size neighborhood of the sample i in the visualization display but not in the original data space. The measure of trustworthiness of visualization can be calculated in the following way:

$$M_1(k) = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in U_k(i)} (r(i,j) - k), \tag{3}$$

where $r(i,j)$ denotes the ranking of the objects in the input space.

The projection onto a lower dimensional output space is said to be *continuous* [9,17] if points near to each other in the original space are also nearby in the output space. Denote $V_i(k)$ the set of those data points that belong to the k-neighbors of data sample i in the original space, but not in the visualization. The measure of continuity of visualization is calculated by the following equation:

$$M_2(k) = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in V_k(i)} (\widehat{r}(i,j) - k), \tag{4}$$

where $\widehat{r}(i,j)$ is the rank of the data sample i from j in the output space.

3.2 Analysis of the Methods

In this subsection the local and global mapping qualities of GNLP-NG, TRN-Map and its combination are analyzed. The presentation of the results comes true through the well known wine data set coming from the UCI Repository of Machine Learning Databases (http://www.ics.uci.edu). The common parameters of GNLP-NG and TRNMap algorithms were in the simulations set as follows: $t_{max} = 200n$, $\varepsilon_i = 0.3$, $\varepsilon_f = 0.05$, $\lambda_i = 0.2n$, $\lambda_f = 0.01$, $T_i = 0.1n$, $T_f = 0.5n$. The auxiliary parameters of the GNLP-NG algorithm were set as $\alpha_i = 0.3$, $\alpha_f = 0.01$, $K = 2$, and if the influence of the neighborhood size was not analyzed, the values of parameter σ were set as follows: $\sigma_i = 0.7n$, $\sigma_f = 0.1$.

The *wine database* consists of the chemical analysis of 178 wines from three different cultivars in the same Italian region. Each wine is characterized by 13 continuous attributes, and there are three classes distinguished. Figure 1 shows the trustworthiness and the continuity of mappings at different number of codebook vectors ($n = 35$ and $n = 45$). These quality measures are functions of the number of neighbors k. As small k-nn-s the local reconstruction performance of the model is tested, while at larger k-nns the global reconstruction is measured. It can be seen, that the NP_TRNMap and DP_TRNMap methods give better performances at larger k-nn values, furthermore these techniques are much less sensitive to the number of the mapped codebooks than the GNLP-GL method. Opposite this the GNLP-NG technique in most cases gives better performance

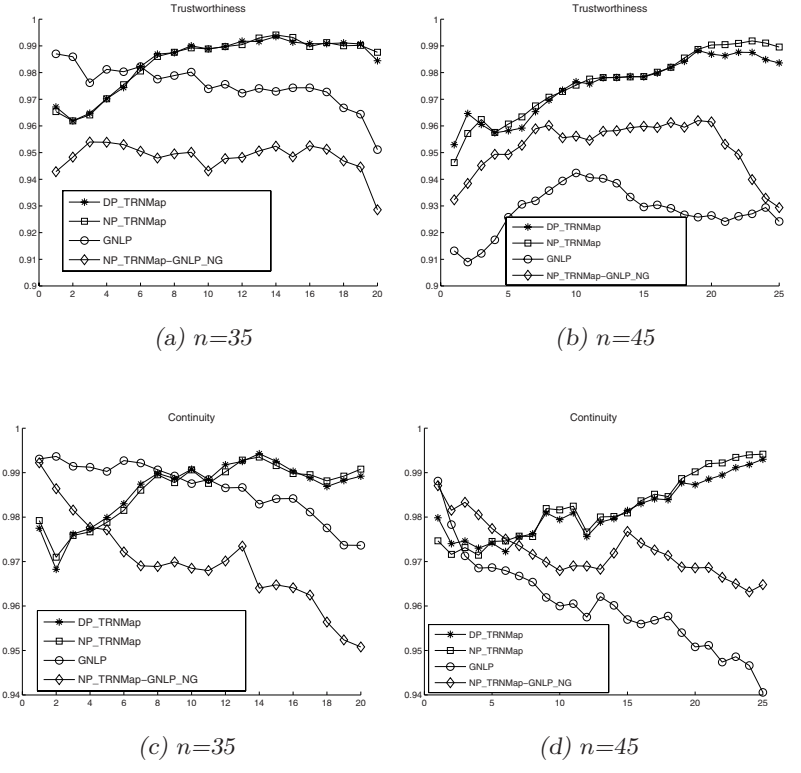


Fig. 1. Trustworthiness and continuity as a function of the number of neighbors k, for the wine data set

at the local reconstruction, and it is sensitive to the number of the neurons. This could be caused by the fact that GNLP-NG applies a gradient based iterative optimization procedure that can be stuck in local minima (e.g. Fig. 1(b)). The GNLP-NG-based fine tuning of the NP_TRNMap improves the local continuity performance of the NP_TRNMap at the expense of the global continuity performance.

Figure 1 shows that GNLP-NG is very sensitive to the number of the codebook vectors. This effect can be controlled by the σ parameter that controls the locality of the GNLP-NG. Figure 2 shows, that the increase of the values σ increases the efficiency of the algorithm. At larger σ the algorithm tends to focus globally, the probability of getting into local minima is decreasing.

The CPU time of different mappings have been also analyzed. The DP_TRNMap and NP_TRNMap require significantly shorter calculation than the GNLP-NG method. The combination of NP_TRNMap with GNLP-NG method decreases the computational time of the GNLP-NG method by a small amount.

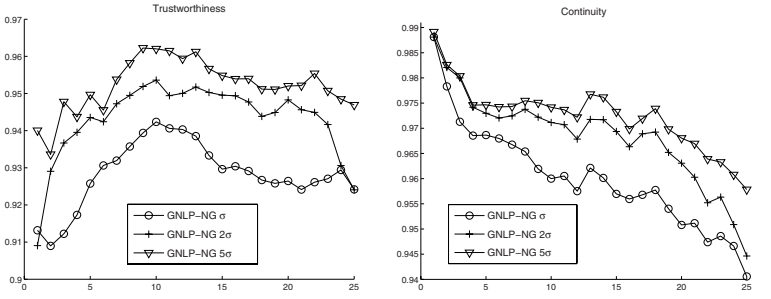


Fig. 2. Trustworthiness and continuity as a function of the number of neighbors k, for the wine data set at different values of σ ($n = 45$)

The mapping methods have also been tested on other benchmark examples, and the results confirmed the previous statements.

4 Conclusion

In this paper we have defined a new class of mapping methods, that are based on the topology representing networks. To detect the main properties of the topology representing network based mapping methods an analysis was performed on them. The primary aim of the analysis was the examination of the preservation of the neighborhood from local and global viewpoint. Both the class of TRNMap methods and the GNLP-NG algorithm utilize neighborhood preservation mapping method, but the TRNMap is based on the MDS technique, while the GNLP-NG utilize an own adaptation rule. It has been shown that: (1) MDS is a global technique, hence it is less sensitive to the number k-nearest neighbors at the calculation of the trustworthiness and continuity. (2) MDS-based techniques can be considered as global reconstruction methods, hence in most cases they give better performances at larger k-nn values. (3) MDS-based techniques are much less sensitive to the number of the mapped codebook vectors than the GNLP-NG technique, which tends to give worse performances when the number of codebook vectors is increased. This could be caused by the fact that GNLP applies a gradient based iterative optimization procedure that can be stuck in local minima. (4) This effect is controlled by parameter σ that influences the locality of the GNLP-NG method. (5) The GNLP-NG-based fine tuning of the MDS-based mapping methods improves the local performance at the expense of the global performance. (6) The GNLP-NG needs more computational time, than the MDS based TRNMap methods. Further research could be the comparison of ViSOM and the proposed TRNMap methods.

Acknowledgement. The authors acknowledge the financial support of the Cooperative Research Centre (VIKKK, project 2004-I), the Hungarian Research

Found (OTKA 49534), the Bolyai Janos fellowship of the Hungarian Academy of Science, and the Öveges fellowship.

References

1. Bernstein, M., de Silva, V., Langford, J.C., Tenenbaum, J.B.: Graph approximations to geodesics on embedded manifolds. Techn. Rep., Stanford Univ. (2000)
2. Borg, I., Groenen, P.: Modern Multidimensional Scaling: Theory and Applications. Springer Series in Statistics. Springer, New York (1997)
3. Estévez, P.A., Figueroa, C.J.: Online data visualization using the neural gas network. Neural Networks 19(6), 923–934 (2006)
4. Estévez, P.A., Chong, A.M., Held, C.M., PerezC., A.: Nonlinear Projection Using Geodesic Distances and the Neural Gas Network. In: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4131, pp. 464–473. Springer, Heidelberg (2006)
5. Hebb, D.O.: The organization of behavior. John Wiley and Son Inc., New York (1949)
6. Hoteling, H.: Analysis of a complex of statistical variables into principal components. Journal of Education Psychology 24, 417–441 (1933)
7. Jolliffe, I.T.: Principal Component Analysis. Springer, New York (1996)
8. Kohonen, T.: Self-Organising Maps, 2nd edn. Springer, Berlin (1995)
9. Kaski, S., Nikkilä, J., Oja, M., Venna, J., Törönen, P., Castrén, E.: Trustworthiness and metrics in visualizing similarity of gene expression. BMC Bioinf. 4(48) (2003)
10. MacQueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
11. Martinetz, T.M., Schulten, K.J.: A neural-gas network learns topologies. In: Kohonen, T., Mäkisara, K., Simula, O., Kangas, J. (eds.) Artificial Neural Netw., pp. 397–402 (1991)
12. Martinetz, T.M., Shulten, K.J.: Topology representing networks. Neural Networks 7(3), 507–522 (1994)
13. Muhammed, H.H.: Unsupervised Fuzzy Clustering Using Weighted Incremental Neural Networks. International Journal of Neural Systems 14(6), 355–371 (2004)
14. Sammon, J.W.: A Non-Linear Mapping for Data Structure Analysis. IEEE Trans. on Computers C18(5), 401–409 (1969)
15. Si, J., Lin, S., Vuong, M.-A.: Dynamic topology representing networks. Neural Networks 13, 617–627 (2000)
16. Vathy-Fogarassy, A., Kiss, A., Abonyi, J.: Topology Representing Network Map–A new Tool for Visualization of High-Dimensional Data. In: LNCS Transactions on Computational Science (to appear)
17. Venna, J., Kaski, S.: Local multidimensional scaling. Neural Networks 19, 889–899 (2006)
18. Yin, H.: ViSOM–A novel method for multivariate data projection and structure visualisation. IEEE Trans. on Neural Networks 13, 237–243 (2002)

The Outer Impartation Information Content of Rules and Rule Sets

Dan Hu[1] and Yuanfu Feng[2]

[1] Beijing Normal University, College of Information Science and Technology
100875 Beijing, China
hufengdd@163.com
[2] Beijing Union University, Basic Courses Department
Beijing 100101, China
longlonghd@163.com

Abstract. The appraisement of rules and rule sets is very important in data mining. The information content of rules is discussed in this paper and is categorized into inner mutual information and outer impartation information. We put forward the viewpoint that the outer impartation information content of rules and rule sets can be represented by relations from input universe to output universe. Then, the interaction of rules in a rule set can be represented by the union and intersection of binary relations expediently. Based on the entropy of relations, the outer impartation information content of rules and rule sets are well measured. Compared with the methods which appraise rule sets by their overall performance (accuracy, error rate) on the given test data sets, the outer impartation information content of rule sets is more objective and convenient because of the absence of test data sets.

Keywords: interestingness measure of rules, relations, outer impartation information content of rules, outer impartation information content of rule sets.

1 Introduction

The assessment of discovered knowledge has become a key problem in the domain of knowledge discovery by the emergence of enormous algorithms for knowledge acquisition. Rule is an important form of discovered knowledge, and interestingness measure is a common tool for the evaluation of it. In these thirty years, there have been many literatures focused on interestingness measures[1,2,3]. The interestingness measures can be divided into objective measures and subjective measures based on the estimator, a computer or human user. The subjective measures, such as Unexpectedness[4,5]and actionability[6], evaluate the rules by the synthesis of cognition, domain knowledge, individual experiences [7]. Variously, the objective measures, such as Coverage, Support, Accuracy and Credibility[7,8],etc, only depend on the structure of a pattern and the underlying data used in the discovery process. Detailedly, the objective interestingness can be categorized into

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 567–577, 2007.

some groups with the criterions and the theories for evaluation. The main criterions for the classification are Correctness[9], Generality[10], Uniqueness[11], and Information Richness[12,13]. We focus on Information Richness in this paper. Whichever criterion is used, there is a fact that almost all of the objective interestingness measures are determined by the quantitative information table of rules. Just because of this fact, most objective interestingness measures can not distinguish the rules having the same quantitative information table, even if these rules are obviously different. (The quantitative information table of a rule $r : c \rightarrow d$ is shown as table 1, where $m(c)$ denotes the set of elements which satisfy the condition expressed by c, and $|\cdot|$ denotes the cardinality of a set.)

Table 1. The quantitative information table of r

	d	$\neg d$	Total
c	$\|m(c) \cap m(d)\|$	$\|m(c) \cap m(\neg d)\|$	$\|m(c)\|$
$\neg c$	$\|m(\neg c) \cap m(d)\|$	$\|m(\neg c) \cap m(\neg d)\|$	$\|m(\neg c)\|$
Total	$\|m(d)\|$	$\|m(\neg d)\|$	$\|U\|$

Example 1. Table 2 represents a database S.

Table 2. S=(U, {Temperature, Noise} ∪ {Speed}, f, V)

Temperature	*Noise*	*Speed*	*number of records*
Low	*Low*	*Low*	20
Medium	*Low*	*Medium*	30
Medium	*Normal*	*High*	10
High	*Normal*	*Medium*	10
High	*Normal*	*High*	30

r_1 and r_2 are rules induced from S, where

r_1 : If Temperature is Low and Noise is Low, Then Speed is Low.

r_2 : If Temperature is Low, Then Speed is Low..

The quantitative information table of r_1 is the same as r_2. There is no difference between the interestingness value of r_1 and r_2, if we estimate the interestingness of r_1 and r_2 by the measures which are absolutely decided by the quantitative information table(such as Support, Coverage, Recall, χ^2 measure, J-measure, Yao's interestingness measures, etc.[7,8]). But the fact is that r_2 is more general and interesting than r_1. So another new measure, which is not solely based on the quantitative information table, is needed to estimate the interestingness of rules.

On the other side, although lots of work have been done to discuss the interestingness of single rule, there are few literatures related to the appraisement of rule sets. Rule sets are usually assessed in terms of overall performance

(accuracy, error rate) on the given test data sets. But the overall performance is not objective enough, because the result of evaluation depends on the choice of test data set. Aggregating the interestingness measures of single rule to appraise rule set is not a good way too, because the existed interestingness measures of single rule can not represent the interaction of rules in the rule set.

In this paper, we propose new measures for the estimation of rules and rule sets. At first, we classify the information content of rules to inner mutual information and outer impartation information. Then, the outer impartation information content of rules is represented by relations from input universe to output universe and measured based on the entropy of relations. Different from the objective measures proposed before, this new measure focuses on the corresponding relation between explanatory attributes and class attributes, and evaluates the information that helps human or receptor to make decision. Then, the union and intersection of relations are used to represent the interaction of rules in a rule set. A measure for the estimation of rule sets is well defined based on the information content of rules, and we named it as the outer impartation information content of rule sets.

2 The Outer Impartation Information Content of Rules

Let $S =< U,\ A = \ C \cup D,\ V,\ f >$ be the database. U is the universe. $C = \{C_1,\ C_2, \cdots,\ C_n\}$ is the set of condition-attributes, and D is the set of decision-attributes (suppose that there is only one attribute included in D). $V = (\bigcup_{i=1}^{n} V_{C_i}) \cup V_D$, where $V_{C_i} = \{c_{i1},\ c_{i2},\ \cdots,\ c_{im_i}\}(i = 1, 2 \cdots, n)$, $V_D = \{d_1,\ d_2,\ \cdots,\ d_l\}$. $f:\ U \times A \to V$ is the evaluation function. A classification rule is defined as

$$r:\ \text{If } C_{k_1} \text{ is } c_{k_1 v_1} \text{ and } C_{k_2} \text{ is } c_{k_2 v_2} \text{ and } \cdots \text{and } C_{k_j} \text{ is } c_{k_j v_j},\ \text{Then } D \text{ is } d_v,$$

where $C_{k_1},\ C_{k_2},\ \cdots, C_{k_j} \in C$, $c_{k_t v_t} \in V_{C_{k_t}}$, $t = 1,\ 2,\ \cdots,\ j$, $v_t \in \{1,\ 2,\ \cdots,\ m_{k_t}\}$. For convenience, a classification rule can be abbreviated to the form $r:\ \ c \to d$, where c is the conjunction of condition-attribute values and d is the predicted class.

Concerning the information content of rules, we think that it should be categorized to two classes:

. inner mutual information content: How much information does the antecedent contribute to the consequent. The strength of the connection between the antecedent and consequent of rules.

. outer impartation information content: How much information is conveyed to the receptor and help the receptor to decide which is the predicted class in different conditions.

Symth and Goodman first define J-Measure[13] to measure the information content of rules,

$$J(r) = P(c)(P(d|c)\log(\frac{P(d|c)}{P(d)}) + (1 - P(d|c))\log(\frac{(1 - P(d|c))}{(1 - P(d))})) = P(c)j(d;c). \tag{1}$$

$j(d;c)$ measures the information that the antecedent c contributes to the consequent d[14]. The other part of J-Measure, $P(c)$, can be viewed as a preference for generality or simplicity of the rule $c \to d$. So, we can find the fact that J-Measure does not focus on the outer impartation information but inner mutual information content of rules.

There are other useful measures which use information theory to discuss the interestingness of rules[7]. Such as Normalized Mutual Information, Yao's interestingness and K-Measure. Just as the analysis of J-Measure, all of these measures focus on the inner mutual information content of rules.

To the best of our knowledge, there is no reported work related to measuring the outer impartation information content of rules. In fact, rules help the receptor to construct relations from condition-attributes to class-attributes, and the relations do help users to make decision and represent the outer impartation information content of rules. That is to say, the outer impartation information content of rules can be represented by relations from input universe to output universe. We have defined the entropy of relations in [15] to estimate the information conveyed by relations.

Definition 1. [15] Let U be the finite universe. $X, Y \in \wp(U)$, R is a relation from X to Y. $\forall x \in X$, $R(x) = \{y \in Y | (x,y) \in R\}$. R' is a relation defined by R,

$$R'(x_i) = \begin{cases} R(x_i) & x_i \in R^{-1}(Y) \\ Y & x_i \overline{\in} R^{-1}(Y), x_i \in X \end{cases} \tag{2}$$

The entropy of R restricted on X is denoted by $H(R \downarrow X)$ and defined as follows:

$$H(R \downarrow X) = -\sum_{x_i \in X} \frac{|R'(x_i)|}{\sum_{x_i \in X} |R'(x_i)|} \log \frac{|R'(x_i)|}{|Y|}. \tag{3}$$

The base of logarithm is 2, and $0 \log 0 = 0$.

Based on the entropy of relations, the outer impartation information content of rules can be easily defined and measured.

Definition 2. Let $S =< U,\ A = \ C \cup D,\ V,\ f >$ be the given database. $C = \{C_1,\ \cdots,\ C_n\}$, $V_{C_i} = \{c_{i1},\ \cdots,\ c_{im_i}\}$, D contains only one element, $V_D = \{d_1,\ \cdots, d_l\}$. Suppose that we have induced a rule r_k from S,

$$r_k:\ \text{If } C_{k_1} \text{ is } c_{k_1 v_1} \text{ and } \ C_{k_2} \text{ is } c_{k_2 v_2} \text{ and } \cdots \text{and } C_{k_j} \text{ is } c_{k_j v_j},\ \text{Then } D \text{ is } d_v,$$

where $C_{k_1},\ C_{k_2},\ \cdots, C_{k_j} \in C,\ c_{k_t v_t} \in V_{C_{k_t}},\ t = 1,\ 2,\ \cdots,\ j,\ d_v \in V_D$. A relation R_{r_k} from $\prod_{i=1}^{n} V_{C_i}$ to V_D is defined to represent the outer impartation information content of r_k.

$$R_{r_k} = Re_{r_k} \cup Ru_{r_k}, \tag{4}$$

$$\begin{aligned} Re_{r_k} = \{(< c_{1h_1}, \cdots, c_{k_1 v_1}, \cdots, c_{ih_i}, \cdots, c_{k_2 v_2}, \cdots, c_{k_j v_j}, \cdots, c_{nh_n} >, d_v) | \\ h_1 \in \{1,\ \cdots,\ m_1\}, h_i \in \{1,\ \cdots,\ m_i\}, h_n \in \{1,\ \cdots,\ m_n\}\}, \end{aligned} \tag{5}$$

$$\begin{aligned} Ru_{r_k} = \{(< c_{1h_1}, \cdots, c_{k_1 s_1}, \cdots, c_{ih_i}, \cdots, c_{k_2 s_2}, \cdots, c_{k_j s_j}, \cdots, c_{nh_n} >, d_q) | \\ h_1 \in \{1,\ \cdots,\ m_1\}, h_i \in \{1,\ \cdots,\ m_i\}, h_n \in \{1,\ \cdots,\ m_n\}, \\ c_{k_t s_t} \in V_{k_t}, t \in \{1,\ \cdots,\ j\}, \vee_{t=1}^{j}(c_{k_t s_t} \neq c_{k_t v_t}),\ q = 1, \cdots, l\}, \end{aligned} \tag{6}$$

Re_{r_k} and Ru_{r_k} represent the expanded information and unknown information of rule r_k, and they all relations from $\prod_{i=1}^{n} V_{C_i}$ to V_D.

Then, the outer impartation information content of r_k is measured by $IC(r_k)$,

$$IC(r_k) = Precision(r_k) \cdot H(R_{r_k} \downarrow \prod_{i=1}^{n} V_{C_i}), \tag{7}$$

where $Precision(r_k)$ is the Precision of r_k, $Precision(r_k) = P(d_v| \wedge_{t=1}^{j} c_{k_t v_t})$, and $H(R_{r_k} \downarrow \prod_{i=1}^{n} V_{C_i})$ is the entropy of R_{r_k} restricted on $\prod_{i=1}^{n} V_{C_i}$.

Remark 1. In definition 2, We do not use $H(R_{r_k})$[15] but $H(R \downarrow \prod_{i=1}^{n} V_{C_i})$ to measure the outer impartation information content of r_k, because classification rule has a direction from antecedent to consequent. If r_k is a rule without direction, such as association rule, we should use $H(R_{r_k})$ to substitute $H(R_{r_k} \downarrow \prod_{i=1}^{n} V_{C_i})$ in equation (7).

Remark 2. If you want to use some new measures to substitute one of the terms, $Precision(r_k)$ and $H(R_{r_k} \downarrow \prod_{i=1}^{n} V_{C_i})$ in equation (7), you must take care of your choice. The key of substitution is the trade-off between accuracy and generality. For example, Lift($P(d_v|\wedge_{t=1}^{j} c_{k_t v_t})/p(d_v)$) or Relative Risk($P(d_v|\wedge_{t=1}^{j} c_{k_t v_t})/P(d_v|\neg(\wedge_{t=1}^{j} c_{k_t v_t}))$) can be used to substitute $Precision(r_k)$.

Table 3. The outer impartation information content of r_1 (R_{r_1})

	Temperature	*Noise*	*Speed*
Original form of r_1	*Low*	*Low*	*Low*
Expanded Information(Re_{r_1})	*Low*	*Low*	*Low*
Unkonwn Information (Ru_{r_1})	*Low*	*Normal*	$\{L,\ M,\ H\}$
	Medium	*Low*	$\{L,\ M,\ H\}$
	Medium	*Normal*	$\{L,\ M,\ H\}$
	High	*Low*	$\{L,\ M,\ H\}$
	High	*Normal*	$\{L,\ M,\ H\}$

Table 4. The outer impartation information content of r_2 (R_{r_2})

	Temperature	*Noise*	*Speed*
Original form of r_2	*Low*		*Low*
Expanded Information (Re_{r_2})	*Low*	*Normal*	*Low*
	Low	*Low*	*Low*
Unkonwn Information(Ru_{r_2})	*Medium*	*Low*	$\{L,\ M,\ H\}$
	Medium	*Normal*	$\{L,\ M,\ H\}$
	High	*Low*	$\{L,\ M,\ H\}$
	High	*Normal*	$\{L,\ M,\ H\}$

Table 5. $J(*)$ and $IC(*)$

	r_1	r_2	Result
J(*)	$\frac{1}{5}\log 5$	$\frac{1}{5}\log 5$	r_1 is the same as r_2
$IC(*)$	$\frac{1}{16}\log 3$	$\frac{1}{7}\log 3$	r_2 is better than r_1

Example 1 (continued). Compare r_1 and r_2 by J-Measure and $IC(*)$. The The outer impartation information content of r_1 and r_2 are shown in table 3 and table 4, where $\{L,\ M,\ H\}$ represents that the corresponding antecedent can lead to Low, Medium or high speed.

$IC(*)$ can distinguish r_1 and r_2, because the outer impartation information content of r_1 is less than r_2.

Proposition 1. $S =< U,\ A = \ C \cup D,\ V,\ f >$ is the given database, $C = \{C_1,\ \cdots,\ C_n\}, V_{C_i} = \{c_{i1},\ \cdots,\ c_{im_i}\}, V_D = \{d_1, \cdots, d_l\}$. r_k is the rule induced from S,

$$r_k:\ \text{If } C_{k_1} \text{ is } c_{k_1 v_1} \text{ and } \ C_{k_2} \text{ is } c_{k_2 v_2} \text{ and } \cdots \text{and } C_{k_j} \text{ is } c_{k_j v_j},\ \text{Then } D \text{ is } d_v.$$

Let $MIN = \min\{m_{l_1}, \cdots, m_{l_{n-j}}\},\ (l_1, \cdots, l_{n-j} \in \{1, \cdots, n\} \setminus \{k_1, \cdots, k_j\})$. $K = \frac{\prod_{i=1}^{n} m_i}{\prod_{t=1}^{j} m_{k_t}}$,

(1). r_{k_s} is specialization of r_k. If the Precision of r_k satisfies the following inequality

$$Precision(r_k) = P(d_v | \wedge_{t=1}^{j} c_{k_t v_t}) \geqslant \frac{K + (\prod_{i=1}^{n} m_i - K) \times l}{K + (MIN \cdot \prod_{i=1}^{n} m_i - K) \times l}, \tag{8}$$

Then, $IC(r_{k_s}) \leq IC(r_k)$.

(2). Suppose r_{k_g} is the generalization of r_k, which is gotten by deleting the condition "C_{k_1} is $c_{k_1 v_1}$" from r_k

$$r_{k_g}:\ \text{If } C_{k_2} \text{ is } c_{k_2 v_2} \text{ and } \ C_{k_3} \text{ is } c_{k_3 v_3} \text{ and } \cdots \text{and } C_{k_j} \text{ is } c_{k_j v_j},\ \text{Then } D \text{ is } d_v.$$

If

$$Precision(r_{k_g}) \leq Precision(r_k) \cdot \frac{m_{k_1} \cdot K + l \cdot \prod_{i=1}^{n} m_i - m_{k_1} \cdot K \cdot l}{m_{k_1} \cdot K + m_{k_1} \cdot l \cdot \prod_{i=1}^{n} m_i - m_{k_1} \cdot K \cdot l}, \tag{9}$$

then

$$IC(r_{k_g}) \leq IC(r_k). \tag{10}$$

Remark 3. When we want to use the way of specialization to capture more information, we should use proposition 1 to estimate if there is a chance to increase the outer impartation information content. On the other side, we always generalize a rule to make it suit for more situation. But we can't generalize the rule blindly. Otherwise, the outer impartation information may be lost in the process of generalization because of the decrease of Precision. Proposition 1 shows us there's a boundary for the Precision of r_{k_g}.

3 The Outer Impartation Information Content of Rule Sets

In section 2, the outer impartation information of a rule is represented by relations. Then, by the union and intersection of relations, we can deal with the interaction of rules in a rule set.

Definition 3. Suppose that we have induced a rule set $Q = \{r_k | k = 1, \cdots, h.\ h \geq 2\}$ from the database S. The expanded information of rule set Q can be defined as

$$Re_Q = \bigcup_{k=1}^{h} Re_{r_k}, \tag{11}$$

The unknown information of rule set Q can be defined as

$$Ru_Q = \bigcap_{k=1}^{h} Ru_{r_k}. \tag{12}$$

Thus the outer impartation information content of Q is represented by R_Q,

$$R_Q = Re_Q \bigcup Ru_Q. \tag{13}$$

$\forall \sigma \in R_Q$, we define the Precision of σ under Q and the Precision of Q as

$$Precision_Q(\sigma) = \max\{Precision(r_k) | r_k \in Q,\ \sigma \in Re_{r_k}\}. \tag{14}$$

$$Precision(Q) = \frac{\sum_{\sigma \in Re_Q} Precision_Q(\sigma)}{|Re_Q|}. \tag{15}$$

The outer impartation information content of rule set Q is measured by $IC(Q)$,

$$IC(Q) = Precision(Q) \cdot H(R_Q \downarrow \prod_{i=1}^{n} V_{C_i}). \tag{16}$$

Remark 4. If $Q = \{r_0\}$, then$IC(Q) = IC(r_0)$.

Remark 5. When there is $'\vee'$ exiting in the antecedent of a rule, we can transfer the rule to a rule set and deal with it by definition 3.

Example 1 (Continued). Suppose $Q = \{r_3,\ r_4\}$ is a rule set induced from database S, where

r_3 : If Noise is Low, Then Speed is Low.,

r_4 : If Temperature is Medium, Then Speed is Medium..

The outer impartation information content of Q is shown in table 6.

$$Precision(Q) = (0.4 + 0.4 + 0.4 + 0.75 + 0.75)/5 = 0.54;$$

Table 6. The outer impartation information content of $Q(R_Q)$

	Temperature	*Noise*	*Speed*
Original Form	*Medium*	*Low*	*Low* *Medium*
Expanded Information	*Low*	*Low*	*Low*
	Medium	*Low*	*Low*
	High	*Low*	*Low*
	Medium	*Normal*	*Medium*
	Medium	*Low*	*Medium*
Unknown Information	*Low*	*Normal*	$\{L,\ M,\ H\}$
	High	*Normal*	$\{L,\ M,\ H\}$

$$IC(Q) = Precision(Q) \cdot (\frac{3}{10}\log\frac{3}{1} + \frac{1}{10}\log\frac{3}{2} + \frac{1}{10}\log\frac{3}{3}) = 0.2884.$$

Definition 4. Let r_k and r_N be rules induced from database S. Let $W = \prod V_{C_i}$, $R \subseteq W \times V_D$. The antecedents of the elements in R is denoted by $R|_W$ and defined as

$$R|_W = \{\omega \in W | \exists d_v \in V_D,\ (\omega,\ d_v) \in R\}.$$

r_N is independent of r_k if and only if

$$(Re_{r_N}|_W) \cap (Re_{r_k}|_W) = \emptyset. \tag{17}$$

Proposition 2. Let $W = \prod V_{C_i}$, $|V_D| = l$. Suppose that we have induced a rule set $Q = \{r_k | k = 1, \cdots, h.\ h \geq 2\} (\neq \emptyset)$ from the database S. Q satisfies

$$\forall r_k \in Q, \quad IC(r_k) \neq 0. \tag{18}$$

then

(1). $$0 \leq IC(Q) \leq \log l. \tag{19}$$

If $|R_Q| = |W| \cdot l$, then $IC(Q) = 0$. If $R_Q|_W = W$, $|R_Q| = |W|$ and $Precision(Q) = 1$, then $IC(Q) = \log l$ holds.

(2). If all rules in Q are independent from each other, then

$$IC(Q) > \sum_{i=1}^{m} IC(r_k). \tag{20}$$

Remark 6. In equation (19), the information content of a rule set may be zero. When we put rules together, the right result seems to be

$$IC(Q) \geq \max\{IC(r_1), \cdots, IC(r_h)\}. \tag{21}$$

But equation (21) is wrong at some time. There may be inconsistent information among the expanded information of rules. In table 6, there is inconsistency in the expanded information of Q:

$$\{(< Medium, Low >, Low), (< Medium, Low >, Medium)\}.$$

Based on the definition of the entropy of relations, it is easily seen that Adding a rule which has conflict with the former rule set will reduce the effective information.

Remark 7. The outer impartation information content of a rule set may be bigger than the sum of the outer impartation information content of each rule in the rule set.

When we estimate the relation between $IC(Q)$ and $\sum_{i=1}^{h} IC(r_k)$, the right result seems to be

$$IC(Q) \leq \sum_{i=1}^{h} IC(r_k), \tag{22}$$

But equation (23) does not always hold, and at sometime, we have

$$IC(Q) > \sum_{i=1}^{h} IC(r_k), \tag{23}$$

just as the result in proposition 2. This result is too strange to be trusted, but it is true. We can use an example which is more direct to explain this phenomenon. Suppose that we want to decide the occurrence of an event E_1. There are two relative evidences e_1 and e_2,

$$e_1: \quad \text{an event } E_2 \text{ has occurred,}$$

$$e_2: \quad E_2 \text{ must occur with } E_1.$$

If we want to solely use e_1 or e_2 to decide the occurrence of E_2, we'll get no information and can't reach the decision. But when we use the evident set $\{e_1,\ e_2\}$, the result is obvious.

The strength of collectivity is much bigger than the sum of individual's at sometime. This is a well known theory and is also held by the society of rules. Different rules interact with each other and remedy each other. This is just the reason of the result shown in equation (23).

4 Experiment

The outer impartation information content(OIIC) of rule sets can be easily incorporated with knowledge discovery algorithms to appraise and compare them without test data sets. The algorithm with maximal OIIC value is the best.

The well-known iris data set proposed by R.A.Fisher is used to compare the function of accuracy and OIIC measure of rule set. In [16], Iris database was transferred to a discrete database by fuzzy partition. We use different measures(SIG,Add,J-measure,Natual Order,ICR) to learn maximal structure rules for the data after discretization. 'Accuracy' and 'OIIC' are used to appraise the rule sets induced by different measures. In table 7, the accuracy of the induced rule set is estimated by Leaving-one-out method.

Table 7. The comparison of accuracy and OIIC measure of rule set

	SIG	$Add\ Value$	$J-measure$	$Natual\ Order$	ICR
$Accuracy$	0.8529	0.9412	0.8919	0.9118	0.9611
$OIIC\ \ value$	0.8389	0.8721	0.8476	0.8945	1.1767

The appraisement result of accuracy and OIIC are

$$ICR > Add\ Value > Natural\ order > J-measure > SIG$$

and

$$ICR > Natural\ order > Add\ Value > J-measure > SIG$$

respectively, where the meaning of '>' is 'better than'. We find the fact that the appraisement result of accuracy is almost the same as OIIC. But it is obvious that the information content measure of rule sets is better because we need not to test the rule sets under different test sets, and the results independent of test data sets are more objective.

Acknowledgments. This research is supported in part by National Natural Science Foundation of China (Grant No. 60774049; No. 40672195);Natural Science Foundation of Beijing(Grant No. 4062020); the National 973 Fundamental Research Project of China (Grant No. 2002CB312200) and the Youth foundation of Beijing Normal University.

References

1. Yao, Y.Y., Zhong, N.: An analysis of quantitative measures associated with rules. In: Proceedings of Pacific-Aisa Conference on Knowledge Discovery and Data Mining, pp. 479–488 (1999)
2. Hilderman, R.J., Hamilton, H J.: Knowledge Discovery and Measures of interest. Kluwer Academic Publishers, Dordrecht (2001)
3. Ohsaki, M., Sato, Y.: Investigation of rule interestingness in medical data mining. In: Tsumoto, S., Yamaguchi, T., Numao, M., Motoda, H. (eds.) AM 2003. LNCS (LNAI), vol. 3430, Springer, Heidelberg (2005)
4. Balaji, P., Alexander, T.: Unexpectedness as a measure of interestingness in knowledge discovery. Decision Support Systems 27, 303–318 (1999)
5. Liu, B., Hsu, W.: Post-Analysis of learned rules. In: proceedings of AAAI, pp. 828–834 (1996)
6. Ailberschatz, A., Tuzhilin, A.: What makes patterns interesting in knowledge discovery systems. IEEE Transact on knowledge and Data Engineer 8(6), 970–974 (1996)
7. Ohsaki, M., Sato, Y.: Evaluation of rule interestingness measures with a clinical data set on hepatitis. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, Springer, Heidelberg (2004)
8. Hamilton, H.J., Shan, N.: Machine Learning of credible classifications. In: Proceedings of Australian Conference on Artificial Intelligence, pp. 330–339 (1997)

9. Shapiro, G.S.: Discovery, analysis and presentation of strong Rules. Knowledge Discovery in Databases, AAAI/MIT press, pp. 229–248 (1991)
10. Gago, P., Bento, C.: A metric for selection of the most promising rules. In: Proceedings of European Conference on the Principle of Data Ming and Knowledge Discovery, pp. 19–27 (1998)
11. Zhong, N., Yao, Y.Y.: Peculiarity oriented multi-database mining. In: Proceedings of European Conference on Principles of data Mining and Knowledge Discovery, pp. 136–146 (1999)
12. Hamilton, H.J., Fudger, D.F.: Estimationg DBLearn's potential for knowledge discovery in databases. Computational Intelligence 11(2), 280–296 (1995)
13. Symth, P., Goodman, R.M.: Rule induction using information theory. Knowledge Discovery in Databases, AAAI/MIT press (1991)
14. Shore, J.E., Johnson, R.W.: Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. IEEE transactions on Information Theory 26(1), 26–37 (1980)
15. Hu, D., Li, H.X.: The entropy of relations and a new approach for decision tree learning. In: Wang, L., Jin, Y. (eds.) FSKD 2005. LNCS (LNAI), vol. 3614, pp. 378–388. Springer, Heidelberg (2005)
16. Castro, J.L., Castro-Schez, J.J.: Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert systems. Fuzzy Sets and Systems 101, 331–342 (1999)

An Engineering Approach to Data Mining Projects

Óscar Marbán[1], Gonzalo Mariscal[2], Ernestina Menasalvas[1], and Javier Segovia[1]

[1] Facultad de Informática, Universidad Politécnica de Madrid.
Campus de Montegancedo s/n, 28660 Boadilla del Monte (Madrid), Spain
{omarban,emenasalvas,fsegovia}@fi.upm.es
[2] Universidad Europea de Madrid
gonzalo.mariscal@uem.es

Abstract. Both the number and complexity of Data Mining projects has increased in late years. Unfortunately, nowadays there isn't a formal process model for this kind of projects, or existing approaches are not right or complete enough. In some sense, present situation is comparable to that in software that led to 'software crisis' in latest 60's. Software Engineering matured based on process models and methodologies. Data Mining's evolution is being parallel to that in Software Engineering. The research work described in this paper proposes a Process Model for Data Mining Projects based on the study of current Software Engineering Process Models (IEEE Std 1074 and ISO 12207) and the most used Data Mining Methodology CRISP-DM (considered as a "facto" standard) as basic references.

1 Introduction

In its early days, software development focused on creating programming languages and algorithms that were capable of solving almost any problem type. The evolution of hardware, continuous project planning delays, low productivity, heavy maintenance expenses, and failure to meet user expectations had led by 1968 to the *software crisis* [1].This crisis was caused by the fact that there were no formal methods and methodologies, support tools or proper development project management. The software community realized what the problem was and decided to borrow ideas from other fields of engineering. This was the origin of software engineering (SE). As of then process models and methodologies for developing software projects began to materialize.

Software development improved considerably as a result of the new methodologies. This solved some its earlier problems, and little by little software development grew to be a branch of engineering. This shift means that project management and quality assurance problems are being solved. Additionally, it is helping to increase productivity and improve software maintenance.

The history of knowledge discovery in databases (KDD), now known as Data Mining (DM), is not much different. In the early 90s, when the KDD processing term was first coined [2], there was a rush to develop DM algorithms that were capable of solving all problems of searching for knowledge in data. Apart from developing algorithms, tools were also developed to simplify the application of DM algorithms. From the viewpoint of DM process models, the year 2000 marked the most important milestone. CRISP-DM (*CRoss-Industry Standard Process for DM*)[3] was published.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 578–588, 2007.

While it is true that the number of applied projects in the DM area is expanding rapidly, neither all the project results are in use [4,5,6] nor do all projects end successfully [7,8]. The failure rate is actually as high as 60% [9]. CRISP-DM is the most commonly used methodology for developing DM projects as a "facto" standard.

Are we at the same point as SE was in 1968? Certainly not, but we do not appear to be on a par yet either. Looking at the KDD process and how it has progressed, we find that there is some parallelism with the advancement of software. From this viewpoint, DM project development is defining development methodologies to be able to cope with the new project types, domains and applications that organizations have to come to terms with. Nowadays, SE pay special attention to organizational, management or other parallel activities not directly related to development, such as project completeness and quality assurance. CRISP-DM has not yet been sized for these tasks, as it is very much focused on pure development activities and tasks.

This paper is moved by the idea that DM problems are taking on the dimensions of an engineering problem. Hence, the processes to be applied should include all the activities and tasks required in an engineering process, tasks that CRISP-DM might not cover. The proposal is inspired by the work done in SE derived from other branches of engineering. It borrows ideas to establish a comprehensive process model for DM that improves and adds to CRISP-DM. Further research will be needed to define methodologies and life cycles, but the basis of a well-defined process model will be there.

2 Data Mining Process Models

There is some confusion about the terminology different authors use to refer to process and methodology.

A process model is defined as the set of tasks to be performed to develop a particular element, as well as the elements that are produced in each task (outputs) and the elements that are necessary to do a task (inputs) [10]. The goal of a process model is to make the process repeatable, manageable and measurable (to be able to get metrics).

Methodology can be defined as the instance of a process model that both lists tasks, inputs and outputs and specifies how to do the tasks [10]. Tasks are performed using techniques that stipulate how they should be done. After selecting a technique to do the specified tasks, tools can be used to improve task performance.

Finally, the life cycle determines the order in which each activity is to be done [11]. A life cycle model is the description of the different ways of developing a project.

From the viewpoint of the above definitions, what do we have in DM? Does DM have process models and/or methodologies? The KDD process [12] has a process model component because it establishes all the steps to be taken to develop a DM project, but it is not a methodology because its definition does not set out how to do each of the proposed tasks. It is also a life cycle. Like the KDD process, Two Crows [13] is a process model and waterfall life cycle. At no point does it set out how to do the established DM project development tasks. SEMMA [14] is the methodology that SAS proposed for developing DM products. Although it is a methodology, it is based on the technical part of the project only. Like the above approaches, SEMMA also sets out a

waterfall life cycle, as the project is developed through to the end. 5 A's [15] is a process model that proposes the tasks that should be performed to develop a DM project and was one of CRISP-DM's forerunners. Therefore, their philosophy is the same: it proposes the tasks but at no point suggests how they should be performed. The life cycle is similar to the one proposed in CRISP-DM. Data Mining Industrial Engineering [16] is a methodology because it specifies how to perform the tasks to develop a DM project in the field of industrial engineering. It is an instance of CRISP-DM, which makes it a methodology, and it shares CRISP-DM's associated life cycle. Finally, CRISP-DM [3] states which tasks have to be carried out to successfully complete a DM project, making it a process model. It is also a waterfall life cycle. CRISP-DM also has a methodological component, as it gives recommendations on how to do some tasks. However, it just proposes other tasks, giving no guidance about how to do them. Therefore, we class CRISP-DM as a process model.

3 Software Engineering Process Models

The SE panorama is quite a lot clearer, and there are two well-established process models: IEEE 1074 [17] and ISO 12207 [18] . In the following, we will analyze both processes in some detail and propose a generic joint process model. This joint model will then be used for comparison with and, if necessary, to expand the CRISP-DM.

3.1 IEEE STD 1074

The IEEE Std 1074 [17] specifies the processes for developing and maintaining software. IEEE Std 1074 neither defines nor prescribes a particular life cycle. Each organization using the standard should instantiate the activities specified in the standard within its own development process.Next, the key processes defined in this process model will be described. The *software life cycle selection process* identifies and selects a life cycle for the software under construction. The *project management processes* are the set of processes that establish the project structure, and coordinate and manage project resources throughout the software life cycle. *Development-oriented processes* start with the identification of a need for automation. With the support of the integral process activities and under the project management plan, the development processes produce software (code and documentation) from the statement of the need. Finally, the activities for installing, operating, supporting, maintaining and retiring the software product should be performed. *Integral processes* are necessary to successfully complete the software project activities. They are enacted at the same time as the software development-oriented activities and include activities that are not related to development. They are used to assure the completeness and quality of the project functions.

3.2 ISO 12207

ISO 12207 divides the activities that can be carried out during the software life cycle into primary processes, supporting processes and organizational processes.

The *primary life cycle processes* are a compendium of processes that serve the primary parties throughout the software life cycle. A primary party is the party that starts or enacts software development, operation or maintenance.

The *supporting life cycle processes* support other processes as an integral part with a distinct purpose and contribute to the success and quality of the software project. The supporting processes are divided into subprocesses, which can be used in other processes defined by ISO 12207. The supporting processes are used at several points of the life cycle and can be enacted by the organization that uses them. The organization that uses and enacts a supporting process manages that process at project level as per the management process, establishes an infrastructure for the process as per the infrastructure process and drives the process at the organizational level as per the improvement process.

The *organizational life cycle processes* are used by an organization to perform organizational functions, such as management, personnel training or process improvement. These processes help to establish, implement and improve software process, achieving a more effective organization. They tend to be enacted at the corporate level and are outside the scope of specific projects and contracts.

3.3 Unification of IEEE STD 1074 and ISO 12207

Having reviewed IEEE Std 1074 and ISO 12207, the goal is to build a joint process model that is as generic as possible to then try to use it as a basis for defining a process model against which to compare CRISP-DM.

If we compare both models, clearly most of the processes proposed in IEEE Std 1074 match up with ISO 12207 processes and vice versa. To get a joint process model we have merged IEEE Std 1074 and ISO 12207 processes. The process selection criterion was to select the most thoroughly defined IEEE Std 1074 and ISO 12207 processes and try not to merge processes from different groups in different process models. According to this criterion, we selected IEEE Std 1074 as a basis, as its processes are more detailed. Additionally, we added the ISO 12207 *acquisition* and *supply processes*, because IEEE Std 1074 states that ISO 12207 acquisition and supply processes should be used [17] if it is necessary to acquire or supply software.

Figure 1 shows the joint process model developed after studying IEEE Std 1074 and ISO 12207 according to the above criteria. Figure 1 also shows the details of the major process groups, the activities they each involve according to the selected standard for that process group. In the next section we will analyse which of the above activities CRISP-DM includes and which it does not in order to try to build a process model for DM projects.

4 SE Process Model vs. CRISP-DM

This section presents a comparison between CRISP-DM and the joint process model discussed in section 3.3. This comparison should identify what SE model elements are applicable to DM projects and are not covered by CRISP-DM. This way we will be able to build a process model for DM projects based on fairly mature SE process models.

PROCESS	ACTIVITY
Acquisition	
Supply	
Software life cycle selection	Identify available software life cycles
	Select software life cycle
Project management processes	
Initiation	Create software life cycle process
	Allocate project resources
	Perform estimations
	Define metrics
Project monitoring and control	Manage risks
	Manage the project
	Retain records
	Identify software life cycle process improvement needs
	Collect and analyze metric data
Project planning	Plan evaluations
	Plan configuration management
	Plan system transition
	Plan installation
	Plan documentation
	Plan training
	Plan project management
	Plan integration
Development-oriented processes	
Pre-development	
Concept exploration	Identify ideas or needs
	Formulate potential approaches
	Conduct feasibility studies
	Refine and finalize the idea or need
System allocation	Analyze functions
	Decompose system requirements
	Develop system architecture
Software importation	Identify imported software requirements
	Evaluate software import sources
	Define software import method
	Import software
Development	
Requirements	Define and develop software requirements
	Define interface requirements
	Priorizate and integrate software requirements

PROCESS	ACTIVITY
Design	Perform architectural design
	Design data base
	Design interface
	Perform detailed design
Implementation	Create executable code
	Create operating documentation
	Perform integration
Post-Development	
Installation	Distribute software
	Install software
	Accept software in operational environment
Operation and support	Operate the system
	Provide technical asistanse and consulting
	Maintain support request log
Maintenance	Identify software improvement needs
	Implement problem reporting method
	Maintenance support request log
Retirement	Notify user
	Conduct parallel operations
	Retire system
Integral processes	
Evaluation	Conduct reviews
	Create traceability matrix
	Conduct audits
	Develop test procedures
	Create test data
	Execute test
	Report evaluation results
Software configuration management	Develop configuration identification
	Perform configuration control
	Perform status accounting
Documentation development	Implement documentation
	Produce and distribute documentation
Training	Develop training materials
	Validate the training program
	Implement the training program

Fig. 1. Joint process model

Note that the correspondence between CRISP-DM and SE process model elements is not exact. In some cases, the elements are equivalent but the techniques are different, whereas, in others, the elements have the same goal but are implemented completely differently.

4.1 Life Cycle Selection Process

The purpose of the set of processes for selecting the life cycle (*Life cycle selection*) in projects is to *select a life cycle* for the project that is to be developed. Based on the type of product to be developed and the project requirements, life cycle models are identified and analysed and a model that provides proper support for the project is selected. This set of processes also extends to third party software *acquisition* and *supply*. These two processes cover all the tasks related to supply or acquisition management. CRISP-DM does not include any of the *acquisition* or *supply* processes at all. DM project development experience suggests that acquisition and supply processes may be considered necessary and third parties engaged to develop or create DM models for projects of some size or complexity. Developers undertaking a DM project also need to *select a life cycle*, and this depends on the type of project to be developed. Life cycle models are used for software development because not all projects are equal, neither do all developers and clients have the same needs. This also applies to DM projects, as a typical client segmentation, is quite a different kettle of fish from predicting aircraft faults. *Life cycle selection* is not an easy task, as you have to take into account the project type in terms of complexity, experience in the problem domain, knowledge of the data that are

being analysed, variability, and data expiration. Therefore, the life cycle selection process is considered useful for DM projects. However, DM project life cycles will have to be defined, as no thorough studies on possible cycles for use or the variables or criteria that distinguish one life cycle from another have yet been conducted.

4.2 Project Management Processes

The set of processes defined here establish the project structure, and coordinate and manage project resources. The project *initiation* process defines the activities for creating and updating the project development or maintenance infrastructure . *Project planning* covers all the processes related to planning project management activities, including contingency planning. The *project monitoring and control process* analyses technical, economic, operational, and timetable risks in order to identify potential problems, and establish the steps for their management. Additionally, it also covers subprocesses related to project metric management.

Project management processes are evidently also necessary when we undertake a DM project. The tasks that are to be performed need to be planned, and there should be a contingency plan because of the high risk involved in DM projects. Also it is necessary to analyse project costs, benefits and ROI. Looking at the tasks covered by the CRISP-DM stages, however, only in the *business understanding (BU)* phase do we find tasks that are related to project management. The *identify major iterations* task is comparable to map activities for the selected life cycle, except that the DM project iterations are only roughly outlined as there are no defined DM life cycles. Additionally, the philosophy behind the *experience documentation* task is the same as the *identify software life cycle process improvement needs*. CRISP-DM's *inventory of resources* task accounts for resources allocation, although its tendency is to identify what resources are available rather than allocating resources throughout the project. CRISP-DM does not cover this issue.

The other tasks proposed by CRISP-DM directly match up with the SE process model tasks. And all the tasks that do not appear in CRISP-DM are considered necessary in a DM project. However, CRISP-DM's biggest snag in terms of project management is related to metrics (*Define metrics, retain records, collect and analyze metrics*). For the most part, this can be attributed to the field's immaturity. There is a need to define DM metrics in order to establish costs and deviations throughout project execution. The other major omission is the evaluation component (*Plan evaluations*). CRISP-DM does have a results evaluation stage, but what we are referring to here is process evaluation as a whole. Configuration management (*Plan configuration management*) aims to manage versions, changes and modifications of each project element. CRISP-DM does not cover DM project configuration management, but we believe that, because of the size of current projects and the teams of human resources working together on such projects, it should. Different people generate multiple versions of models, initial data sets, documents, etc., in a project. Therefore, if they are not well located and managed, it is very difficult to go back to earlier versions, should the current versions not be valid, and there is a risk of confusing models, data and documentation for different versions.

Additionally, any DM project should include tasks for managing the transfer and use of the results (*Plan system transition, plan installation*), tasks that CRISP-DM does not

cover either. Finally, the other major oversight, fruit of process immaturity, is the documentation task (*Plan documentation*). Reports are generated in all stages, but there is no task aimed at planning what this documentation should be like to conform to thorough standards. This would improve documentation evaluation and review and facilitate work on process improvement.

4.3 Development-Oriented Processes

Software development-oriented processes start with the identification of a need to automate some tasks for performance using a computer (*Identify ideas or needs*). With the support of the *integral process* activities and subject to the project management plan (*Plan project management*), the *development processes* produce the software. Finally, activities for installing (*Installation*), operating (*Operation and support*), supporting (*Operation and support*), maintaining (*Maintenance*) and retiring (*Retirement*) the software product should be performed. They are subdivided into *pre-development*, *development* and *post-development* processes. DM projects start with the need to gather knowledge from an organization's data to help in business decision-making, knowledge that can be used directly or can be integrated into the organization's systems. This is the most mature set of processes at present, as all the existing "methodologies" for DM project development focus primarily on this part. As for SE, these processes can also be divided into *pre-development*, *development* and *post-development* stages.

Pre-Development are related to everything that you have to do before you start to build the system, such as *concept exploration* or *system allocation requirements*. The *concept exploration* process includes identifying an idea or need (*Identify ideas or needs*) for the system to be developed, and the formulation (*Formulate potential approaches*), evaluation (*Conduct feasibility studies*) and refinement of potential solutions at system level (*Refine and finalize the idea or need*). Once the system limits have been established, a statement of need is generated for the system to be developed. This statement of need starts up the *system allocation* process and/or *requirements* process and feeds the *project management* processes. The statement of need is as necessary in DM projects as in any other project; it is a starting point for project development as it provides an understanding of the problem to be solved. Because of its importance, CRISP-DM already accounts for this process. However, it is spread across different stages and always in the *business understanding* stage at the start of the project. The *software importation* process is related to the reuse of existing software. In the case of software, this process provides the means required to identify what requirements imported software can satisfy and evaluate the software to be used. Software does, in principle, not need to be imported in a DM project, because a DM project gathers knowledge and does not develop software. Its equivalent in a DM project would be to import existing DM models that are useful for the current project. For example, one usual practice is to have a client clustering and use that clustering in the ongoing project to classify clients. Therefore, a process that manages the importation of DM models for use in the ongoing project is also required.

Development is responsible for building the software or gathering knowledge in the case of DM projects. There is no exact match between the development processes in

DM projects and SE projects, as the ends are completely different. DM projects aim to gather knowledge, whereas SE projects target software construction. Even so, they share the same phases: *requirements* definition, solution *design* and solution development (*Implementation*). The requirements stage bears most resemblance, as its aim is to gather the client needs and describe these needs in practical terms for the designers and/or implementers (*Assess situation and Determine DM goals*). As for software, DM's design stage has to design the software support for data, since the available data will ultimately be analysed on the software support. However, the key SE design task, which is *"perform architectural design"*, has no direct equivalent in DM. As already mentioned, the goal of SE design is to translate specifications and requirements into a preliminary design of the solution (i.e, object-oriented design). Therefore, perhaps the best thing would be to equate this task to the early decision made on what DM paradigms (clustering, classification, etc.), are to be explored to achieve the *DM Goals*. This would fit in with the later *implementation* phase, where the modelling technique will be selected (*Select modeling technique*) for each goal. There is no direct mapping between the implementation stages, as the goal they pursue is different. This is the best researched stage of DM, on which all the proposed "methodologies" focus. The implementation stage would be equivalent to gathering and analysing the data available for the project, the creation of new data from what are already available, tailoring for DM algorithms and the creation of DM models, all of which are covered by CRISP-DM.

Post-Development *processes* are the processes that are enacted after the software has been built. The *installation* process implies the transportation and installation of a system from the development environment to the operating environment. The *operation and support* process involves system operation by the user. Support includes technical assistance, user queries and support request entry in the support request log. This process can start up the *maintenance* process that provides feedback information to the software life cycle and leads to changes. Finally, the retirement process is the *retirement* of an existing system by withdrawing it from operation. The knowledge gathered in DM projects should be passed on to the user and installed either as pure knowledge or integrated into the client organization's software system for use. The *operation and support* process is necessary to validate the results and how they are interpreted by the client in a real environment in the same way as the *maintenance* process is required to update models obtained or to discover which of the gathered knowledge is erroneous or invalid when new data are entered. This can lead to backtracking in the global process in order to select new attributes or techniques not considered before. As regards *retirement*, DM models also have a period of validity, as if the data profiles change, the models will also change and will no longer be valid. CRISP-DM neither satisfactorily nor completely covers any of the above processes, despite their importance.

4.4 Integral Processes

Integral processes are necessary to successfully complete the project. They are enacted simultaneously to development processes and include activities that are unrelated to development. They are used to assure the completeness and quality of the project functions. The *evaluation* processes are used to discover defects in the product or in the

process used to develop the project. This process covers the performance of all the verification tasks to assure that all the requirements are satisfied. The *configuration management* process identifies the structure of a system at a given time in the life cycle (called system configuration). Its goal is to control system changes and maintain system coherence and traceability. On the other hand, the *documentation development* process is the set of activities that produce, distribute and maintain the documents developers and users require. Finally, the *training* process includes the development of training programs for staff and clients and the preparation of proper training materials. The *documentation development* process for DM projects will be almost the same as for SE, but changes should be made to how the *evaluation* process is done. The *configuration management* process is an especially important CRISP-DM omission, as mentioned earlier. This process is considered absolutely necessary, because one or more people developing a DM project generate a great many versions of input data, models and documents, etc. If these versions are not properly organized by means of configuration management, it is very difficult to return to previous models if it is necessary. We believe that any new DM process model should account for the *training* process, making a distinction between data miner training and user training. To be able to repeat the process enacted in the project or properly interpret the results when new data become available, users sometimes need to be trained in DM.

5 A Process Model for Data Mining Engineering

Having compared CRISP-DM with a SE process model, we find that many of the processes defined in SE and that are very important for developing any type of DM engineering project are missing from CRISP-DM. What we propose is to take the tasks and processes that CRISP-DM includes and organize them by processes similar to those

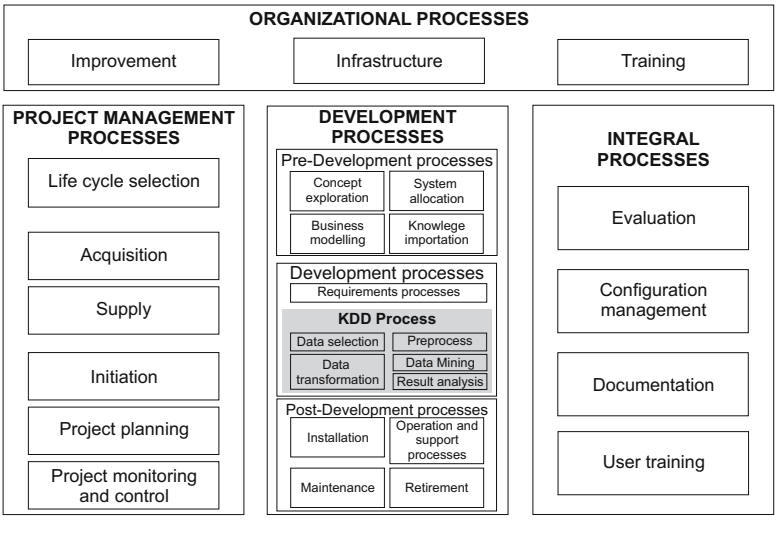


Fig. 2. Data Mining engineering process model

covered in SE and add what we consider to be key development activities. The activities missing from CRISP-DM are primarily *project management* processes, *integral* processes and *organizational* processes. Figure 2 shows an overview of the proposed process model[1], including subprocesses. *KDD process* is the core of *development*.

6 Conclusions

After analysing SE standards, we developed a joint model that we used to compare SE and DM procedures process by process and activity by activity. This comparison highlighted that CRISP-DM either fails to address many tasks related to management, organization and project quality at all or, at least, in enough detail to be able to deal with the complexity of projects now under development. These projects tend to involve not only the study of large volumes of data but also the management and organization of large interdisciplinary human teams. As a result, we proposed a process model for DM engineering that covers those aspects, making a distinction between what is a process model from what is a methodology and life cycle. The proposed process model includes all the activities covered in CRISP-DM, but spread across process groups according to more comprehensive and advanced standards of a better established branch of engineering with over 40 years of experience: SE. The model is not complete, as this paper merely states the need for the subprocesses and especially the activities set out in IEEE Std 1074 or ISO 12207 but missing in CRISP-DM.

References

1. Naur, P., Randell, B.: Software engineering: Report on NATO conference (1969)
2. Piatetsky-Shaphiro, G., Frawley, W.: Knowledge Discovery in Databases. AAAI/MIT Press, MA (1991)
3. Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R.: CRISP-DM 1.0 step-by-step data mining guide. Technical report, CRISP-DM (2000)
4. Eisenfeld, B., Kolsky, E., Topolinski, T.: 42 percent of CRM software goes unused (February 2003), http://www.gartner.com
5. Eisenfeld, B., Kolsky, E., Topolinski, T., Hagemeyer, D., Grigg, J.: Unused CRM software increases TCO and decreases ROI (Febrero 2003), http://www.gartner.com
6. Zornes, A.: The top 5 global 3000 data mining trends for 2003/04. META Group Research-Delta Summary, 2061 (March 2003)
7. Edelstein, H.A., Edelstein, H.C.: Building, Using, and Managing the Data Warehouse. In: Data Warehousing Institute, 1st edn., Prentice Hall PTR, Englewood Cliffs (1997)
8. Strand, M.: The Business Value of Data Warehouses - Opportunities, Pitfalls and Future Directions. PhD thesis, University of Skövde (December 2000)
9. Gondar, J.E.: Metodología Del Data Mining. Data Mining Institute, S.L (2005)
10. Pressman, R.: Software Engineering: A Practitioner's Approach. McGraw-Hill, New York (2005)
11. Moore, J.: Software Engineering Standards: A User's Road Map. IEEE, CA (1998)

[1] This work was conducted as part of the CYCIT-funded project no. TIN2004-05873.

12. Fayyad, U., Piatetsky-Shapiro, G., Smith, P., Uthurusamy, R.: Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, MA (1996)
13. Two Crows Corp. Introduction to Data Mining and Knowledge Discovery. 3rd edn. (1999)
14. SAS Institute. SEMMA data mining methodology (2005), http://www.sas.com
15. de Martínez Pisón, F.J.: Optimización Mediante Técnicas de Minería de Datos Del Ciclo de Recocido de Una Línea de Galvanizado. PhD thesis, Universidad de La Rioja (2003)
16. Solarte, J.: A proposed data mining methodoloy and its aplication to industrial engineering. Master's thesis, University of Tennessee, Knoxville (2002)
17. IEEE. Standard for Developing Software Life Cycle Processes. IEEE Std. 1074-1997. IEEE Computer Society, Nueva York (EE.UU.) (1991)
18. ISO. ISO/IEC Standard 12207:1995. Software Life Cycle Processes. International Organization for Standarization, Ginebra (Suiza) (1995)

Classifying Polyphonic Melodies by Chord Estimation Based on Hidden Markov Model

Yukiteru Yoshihara[1], Takao Miura[1], and Isamu Shioya[2]

[1] Dept.of Elect.& Elect. Engr., HOSEI University
3-7-2 KajinoCho, Koganei, Tokyo, 184–8584 Japan
[2] Dept.of Informatics, SANNO University
Kamikasuya 1573, Isehara, Kanagawa, Japan

Abstract. In this investigation we propose a novel approach for classifying polyphonic melodies. Our main idea comes from for automatic classification of polyphonic melodies by *Hidden Markov model* where the states correspond to well-tempered chords over the music and the observation sequences to some feature values called *pitch spectrum*. The similarity among harmonies can be considered by means of the features and well-tempered chords. We show the effectiveness and the usefulness of the approach by some experimental results.

Keywords: Melody Classification, Melody Features, Markov process, Hidden Markov Modeling.

1 Motivation

We propose a novel approach for classifying polyphonic melodies. We define similarity among polyphonies in terms of *features*, and generate Hidden Markov Model (HMM) analyzing training music data and classifying any music with *Viterbi* algorithm. During the process, we estimate chord progression since state transition corresponds to chord progression in principle. Readers are assumed to be familiar with basic notions of music[5] and basic IR techniques[4].

Section 2 and 3 contain several definitions of features for melody description that have been proposed so far. In section 4 we review Hidden Markov models and discuss how to apply HMM to our issue of music classification. We show some experiments and some relevant works in section 6. Finally we conclude our investigation in section 7.

2 Melody and Polyphony

Music consists of many *tones*. Each tone consists of *pitch*, *duration* and *strength*. Pitch means modulation or height of tone that is defined by *frequency*. For example, 440Hz sounds like *A* (*la*). Two times height is called an *octave* thus 880Hz sounds like *a*, an octave higher pitch[1]. An *interval* is a distance between two

[1] We denote an octave pitch by small letter.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 589–598, 2007.

pitch. We equally divide one octave into 12 *semitones*. Given an octave between `A` and `a`, we obtain 12 semitones denoted by `A,A#,B,C,C#,D,D#,E,F,F#,G,G#`. The *well-tempered* sequence consists of 2,1,2,2,1,2,2 semitones or 2,2,1,2,2,2,1 semitones between adjacent pitch. The former sequence is called *minor* and the latter *major*. Also, in the sequence, `A`, `C` are called *keys*. *Transposition* means relative movement of keys while keeping the number of semitones between adjacent pitch of minor/major intervals.

Each tone has its *duration* which describes length aspect of the pitch. *Melody* is a sequence of tones, or pitch/duration while *chord* or *harmony* is description of tones in parallel which are assumed to be played simultaneously. *Theme* is an intended melody which composers think most important. Music except theme is called *accompaniment*. *Polyphony* is music with accompaniment while *monophony* is the one without. That is, in monophony, there can be at most one tone in music. *Five horizontal lines* contain expression which is described by a list of notes with pitch and duration in both sequence and parallel. The expression on the lines is called *score*. Here melody is described by a set of notes arranged into a sequence. *Beat* is a summarized duration in the notes, and a *bar* is a partial description in a score which contains equal size of beat. *Time of signature* means the number of the beat and its characteristic. Especially *rhythm* is a rule how beat and the characteristic is constructed.

3 Features for Melody Classification

To specify and classify melodies, we should examine what kinds of semantics they carry and we should describe them appropriately. Since we need score-based features for classification purpose, we should examine notes over score or in bars. We discuss several kinds of features, and we put these characteristic values into vector spaces[4]. Similarity between two melodies is defined by their *cosine value*. The similarity can be modeled by ranking these values[12].

First of all, let us describe several features for monophonic melody description[8]. *Melody Contour* is one of the major techniques proposed so far[1,7,11]. *Pitch Contour* is the one where we put stress on incremental transition of pitch information in monophonic melody. Unfortunately there exist many problems in pitch contour[11].

Given a melody on score, we introduce *Pitch Spectrum* per bar in the melody for similarity measure[8]. *Pitch Spectrum* is a histogram where each column represents total duration of a note within a bar. The spectrum constitutes a vector of $12 \times n$ dimensions for n octaves range. We calculate pitch spectrum to every bar and construct characteristic vectors prepared for querying music.

By pitch spectrum we can fix several problems against incomplete melody. In fact, the approach improves problems in swinging and grace. Note that score approach improves issues in rhythm, keys, timbre, expression, speed, rendition and strength aspects of music. Some of the deficiencies are how to solve transposition (relative keys) issues and how to distinguish majors from minors[8]. Especially the latter issue is hard to examine because we should recognize the contents.

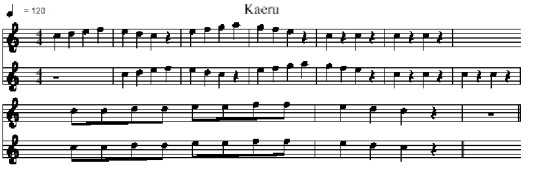


Fig. 1. Score of *A Song of Frogs*

On the features for polyphonic melodies, we assume that polyphonic scores are given in advance to obtain the feature values. Also, we extract all the tones in each bar from polyphonic music and put them into a spectrum in a form of vector. Since the pitch spectrum (for monophonic melody) has been generated as a histogram in which each column represents total duration of a note within the bar, it is straightforward to generate a (polyphonic) pitch spectrum from counting notes appeared in all the parts in the bar. Clearly the new spectrum reflects not only all the tones in the bar but also all the noises for classification like grace/trill notes. We take modulo 12 to all the notes (i.e., we ignore octave). Then we define *(polyphonic) pitch spectrum* as the pitch spectrum as mentioned that consists of only n biggest durations considered as a chord. If there exist more than n candidates, we select the n tones of the highest n pitch. Note that we select n tones as a chord but ignore their explicit duration. And we define the *feature description* of the length m as a sequence $w_1, ..., w_m$ where each feature w_j is extracted from i-th bar of music of interests.

Example 1. Let us describe our running example "A Song of Frogs" in figure 1. Here are all the bars where each collection contains notes with the total duration counted the length of a quarter note as 1.

```
{C:1, D:1, E:1, F:1}, {C:2, D:2, E:2, F:1}, {C:1, D:1, E:2, F:1, G:1, A:1 }, {E:2,
F:2, G:2, A:1}, {C:2, E:1, F:1, G:1}, {C:4}, {C:3, D:1, E:1, F:1}, {C:2, D:2, E:2,
F:1}
```

The sequence of the pitch spectrums constitute the new features for all the bars by top 3 tones. In this case we get the feature description (DEF, CDE, EGA, EFG) for the first 4 bars.

```
Bar1 : {C:1, D:1, E:1, F:1} = {DEF}
Bar2 : {C:2, D:2, E:2, F:1} = {CDE}
Bar3 : {C:1, D:1, E:2, F:1, G:1, A:1} = {EGA}
Bar4 : {E:2, F:2, G:2, A:1} = {EFG}
```

4 Hidden Markov Model

A *Hidden Markov Model* (HMM) is an automaton with output where both the state transition and the output are defined in a probabilistic manner. The state transition arises according to a simple Markov model but it is assumed that we don't know on which state we are standing now[2], and that we can observe an

[2] This is why we say *hidden*.

output symbol at each state. We could estimate the transition sequences through observing output sequence.

Formally a HMM model consists of (Q, Σ, A, B, π) defined below[6]:

(1) $Q = \{q_1, \cdots, q_N\}$ is a finite set of states
(2) $\Sigma = \{o_1, \cdots, o_M\}$ is a finite set of output symbols
(3) $A = \{a_{ij}, i, j = 1, ..., N\}$ is a probability matrix of state transition where each a_{ij} is a probability of the transition at q_i to q_j. Note $a_{i1} + ... + a_{iN} = 1.0$.
(4) $B = \{b_i(o_t), i = 1, ..., N, t = 1, ..., M\}$ is a probability of outputs where $b_i(o_t)$ is a probability of an output o_t at a state q_i
(5) $\pi = \{\pi_i\}$ is an initial probability where π_i is a probability of the initial state q_i

In this work, each state corresponds to a well-tempered chord such as $< C >$ = CEG, and the set of states depends on a polyphonic melody. Output symbols (pitch spectrum such as CDE) should be observable and identifiable in our case. Note spectrums do not always go well with well-tempered chords theoretically.

A HMM is suitable for estimation of *hidden* sequences of states by looking at observable symbols. Given a set of several parameters, we can obtain the state sequence which is the most likely to generate the output symbols. The process is called a *decoding problem* of HMM.

Here we define the most likely sequence of states as the one by which we obtain the highest probability of the output generation during the state transition. The procedure is called *Maximum Likelihood Estimation* (MLE). In the procedure, once we have both sequences of the states and the output symbols, we can determine the probabilities (or *likelihood*) of the state transition and of the output generation along with the state transition. Putting it more specifically, when we have the state transition $q_1 q_2 \cdots q_T$ and the output sequence $o_1 o_2 \cdots o_T$, we must have the likelihood as $\pi_{q_1} b_{q_1}(o_1) \times a_{q_1 q_2} b_{q_2}(o_2) \times \cdots \times a_{q_{T-1} q_T} b_{q_T}(o_T)$. A naive calculation process is called a *Forward* algorithm. *Viterbi* algorithm is another solution for the decoding problem. Given the output sequence $o_1 \cdots o_T$, the algorithm is useful for obtaining the most likely sequence of the states by taking the highest likelihood $\delta_t(j)$ of an output o_t at a state q_i to go one step further to q_j. That is, the algorithm goes recursively as $\delta_{t+1}(j) = \max_i(\delta_t(i) a_{ij}) b_j(o_{t+1})$. During the recursive calculation, we put the state q_j at each time t, and eventually we have the most likelihood sequence $q_1 \cdots q_T$.

In a HMM, there is an importnat issue, how to obtain initial HMM parameters A, B and π. One of the typical approach is *supervised learning*. In this approach, we assume *training data* in advance to calculate the model, but the data should be correctly classified by hands since we should extract typical patterns them by examining them. Another approach comes, called *unsupervised learning*. Assume we can't get training data but a mountain of unclassified data except a few. Once we obtain strong similarity between the classified data and unclassified data (such as high correlation), we could extend the training data in a framework of Expectation Maximization (EM) approach[6]. Here we take supervised learning by analysing scores.

5 HMM for Melody Classification

In this investigation, we assume a collection of music classes and we classify an unknown music d into one of the classes where $d = \{w_1, ..., w_m\}$ and w_j are a feature. In our approach, we consider well-tempered chords[5] as states, and given vectors of pitch spectrums considered as observable sequences, we estimate how the state transition arises and what class of music is most likely.

According to the general procedure of HMM, let us apply the HMM to our problems. We already know a set of well-tempered chords by theory of music[5] and a set of possible features (pitch spectrums). Given a collection $\mathcal{C}$ of polyphonic melodies with classes as training data and an unknown music d to be examined, we estimate a class label c_k of d.

(a) By examining $\mathcal{C}$, we generate a probability matrix A of state transition and probability B of symbol output at each state in a HMM model (Q, Σ, A, B, π) where Q means a set of well-tempered chords, Σ a set of possible features.
(b) Then we estimate a membership probability $P(c_i|d)$ of a class c_i by using the HMM model, $i = 1, ...$ Then, by Maximum Likelihood Estimation (MLE), we have $c_k = ArgMax_{c \in C} P(c|d)$.

Since we classify unknown music into one of the classes given in advance, we compare pitch spectrum with each other, but very often we have many chances to see no common chord among features such as { CDE, DEF } and { CEG, DEG }. Note we have constructed sequence of pitch spectrum in each bar[3]. During comparison of feature descriptions of two music d_1, d_2, it is likely to have some feature (a chord) w in d_1 but not in d_2 at all. In this case, the probability must be 0.0 in d_2 and the membership probability should be zero. Then the two music can't belong to a same class even if the most parts look much alike. Such situation may arise in the case of noises or trills.

To solve this problem[4], we introduce a notion of similarity between each pair of chords and we adjust the probabilities with them. We introduce a notion of *similarity* between two pitch spectrum and we adjust probability of output observation at each state. Assume there are $w_1, ..., w_k$ outputs at a state s, and we see an observation w. We define the probability $P(w_i|s)$ where we have an observation w_i at the state s. Then the observation probability of w at s, denoted by $P'(w|s)$, is adjusted as $P'(w|s) = \sum_i^k P(w_i|s) \times sim(w_i, w)$ where *sim* means *cosine* similarity. Given observations $w_1...w_k$ at states $s_1, ..., s_k$, we see the adjusted probability of the observation sequence $P'(w_1...w_k|s_1...s_k) = \prod_{i=1}^k P'(w_i|s_i)$ as usual.

Example 2. First of all, we show the similarity V of two spectrums $\{CDE\}$ and $\{CDF\}$. Since we have two common tones C, D, $V = \frac{2}{\sqrt{3 \times 3}} = 0.66$.

[3] Remember pitch spectrum is a histogram in which each column represents total duration of a note within a bar and we select top n tones for the spectrum.
[4] Some sort of revisions are usually introduced which causes erroneous classification.

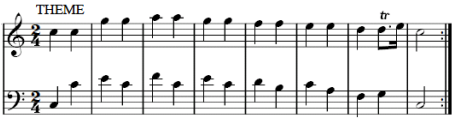


Fig. 2. Mozart Variation KV.265: *Ah, Vous dirai-Je, Maman*

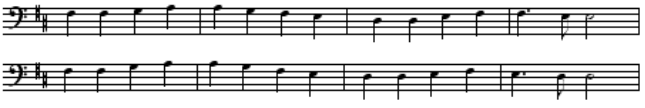


Fig. 3. Beethoven: Op.125

Then let us illustrate our approach as running examples with chords of top 3 tones. We have *A Song of Frog* (d_1) with a label `A` in figure 1, and *Ah, Vous dirai-Je, Maman* in C Major, KV.265, by Mozart (d_2) with a label `B` in figure 2. Let $L_1 = \{d_1, d_2\}$ and we classify Symphony Number 9 (Opus 125) by Beethoven (d_3) in figure 3. Here we assume d_3 contains monophonic melody. We translate the first 8 or 9 bars of these music into `abc` format[13] as follows:

d_1: `{CDEF,-}, {EDC,CDEF}, {EFGA,EDC}, {GFE,EFGA}, {CC,GFE}, {CC,CC}, {C/2C/2D/2D/2E/2E/2F/2F/2,CC}, {EDC,C/2C/2D/2D/2E/2E/2F/2F/2}, {CDE}`

d_2: `{CCGG,CCEC}, {AAGG,FCEC}, {FFEE,DBCA}, {DD3/4E/4C2,FGC}, {CCGG,CCEC}, {AAGG,FCEC}, {FFEE,DBCA}, {DD3/4E/4C2,FGC}`

d_3: `{FFGA}, {AGFE}, {DDEF}, {F3/2E/2E2}, {FFGA}, {AGFE}, {DDEF}, {E3/2D/2D2}`

The sequences of the pitch spectrums constitute the new features for all the bars by the tones. They are *observation sequences* (outputs) as in table 1. In this work, we consider well-tempered chords as states. Then estimation of state transition by looking at observation sequences means *chords progression*, i.e., how chords go along with polyphonic music, based on HMM. To training data, we give the states initially by hands according to the theory of music as illustrated in table 2.

It is possible to count how many times state transition arises between two states and how many output sequences are observed at each state. Eventually we obtain state transition diagrams with probabilities as shown in figures 4(a) (A Song of Frogs) and 4(b) (KV.265). By using the state transition diagrams, we classify d_3. Here we obtain each class membership probability by multiplying each probability of state transition and output in the corresponding diagram, but let us note we should examine extended probabilities at every state. Then we can apply MLE to d_3 through Forward algorithm or Viterbi algorithm.

Algorithm	$P(A\|d_3)$	$P(B\|d_3)$
Forward	4.006×10^{-3}	2.216×10^{-3}
Viterbi	4.006×10^{-3}	1.089×10^{-3}

In any cases, the probability A for d_3 is bigger than B and we should assign Beethoven Op.125 (d_3) to "A Song of Frogs" (A).

Table 1. Tones Appeared and Observation Sequences

Bar	d_1	d_2	d_3
1	{C:1, D:1, E:1, F:1}	{C:5, E:1, G:2}	{F:2, G:1, A:1}
2	{C:2, D:2, E:2, F:1}	{C:2, E:1, F:1, G:2, A:2}	{E:1, F:1, G:1, A:1}
3	{C:1, D:1, E:2, F:1, G:1, A:1}	{C:1, D:1, E:2, F:2, A:1, B:1}	{D:2, E:1, F:1}
4	{E:2, F:2, G:2, A:1}	{C:13/4, D:7/4, E:1/4, F:1, G:1}	{E:5/2 , F:3/2}
5	{C:2, E:1, F:1, G:1}	{C:5, E:1, G:2}	{F:2, G:1, A:1}
6	{C:4}	{C:2, E:1, F:1, G:2, A:2}	{E:1, F:1, G:1, A:1}
7	{C:3, D:1, E:1, F:1}	{C:1, D:1, E:2, F:2, A:1, B:1}	{D:2, E:1, F:1}
8	{C:2, D:2, E:2, F:1}	{C:13/4, D:7/4, E:1/4, F:1, G:1}	{D:5/2, E:3/2}
9	{C:1, D:1, E:1 }		

Bar	1	2	3	4	5	6	7	8	9
d_1	DEF	CDE	EGA	EFG	CFG	C	CEF	CDE	CDE
d_2	C	CEG	CFA	CEG	DFB	CEA	DFG	C	
d_3	FGA	FGA	DEF	EF	FGA	FGA	DEF	EF	

Table 2. State Sequences by Well-Tempered Chord

Bar	1	2	3	4	5	6	7	8	9
d_1	C	C	C	C	C	C	C	C	C
d_2	C	C	F	C	G	C	G	C	

6 Experimental Results

6.1 Preliminaries

We examine three kinds *variations for piano* where a variation consists of themes and its various transformations. We assume all the training data are variations and considered their themes as labels.

Here we examine 3 variations, "Ah, Vous dirai-Je, Maman" in C Major (KV. 265) by Mozart, "Impromptus" in B flat Major (Op.142-3) by Schubert and "6 Variations on theme of Turkish March" in D Major (Op.76) by Beethoven. They contain 12, 5 and 6 variations respectively and 23 variations in total. Note there is no test collection reported so far.

All of 3 themes and 23 variations are processed in advance into a set of feature descriptions. We preprocess the feature descriptions of all the bars of the 3 themes to obtain Hidden Markov Models. Then we calculate the two collections of the feature descriptions, one for the first 4 bar, another for the first 8 bars to all the variations. In this experiment, we examine 3 kinds of chords consisting of the 3, 4 and 5 longest tones. Thus we have $23 \times 2 \times 3 = 138$ features.

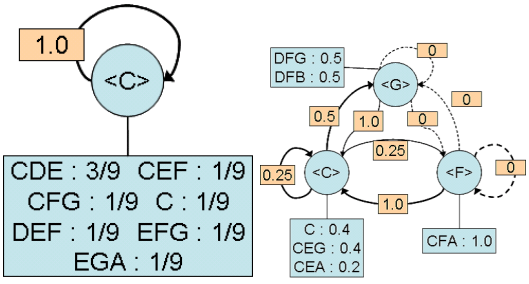


Fig. 4. A Song of Frogs (a) and KV.265 (b) - State Transition Diagrams

Also we give a well-tempered chord to each bar of each training music by hand in advance. Once we complete all the preparations, we examine unlabeled music and guess states (chords) by our approach.

We have 3 classes (labels), *Mozart, Schubert* and *Beethoven* according to the themes. We classify all the 23 unlabeled melodies into one of the 3 labels. We say a variation is *correctly* classified if the music is composed by the label person. Formally the correctness ratio is defined as $p/23$ where p means the number of correctly classified melodies. During classification process, we also estimate a well-tempered chord to each bar, since a state in HMM corresponds to a chord (one chord in each bar).

6.2 Results

Let us show the correctness ratio to each theme in table 3 (a), and the correctness ratio to chords (in each theme) estimated by Viterbi algorithm in table 3 (b). In the latter case, we examine, to all of 8 bars, whether the chords are estimated correctly or not.

Looking at the experimental results in tables 3 (a) and 3 (b), the readers can see the perfect results (100.0 % of the correctness ratio) for theme classification with *only* 3 tones of every case of bars. Similarly we get 69.6 % of the correctness ratio (chord) with 3 tones in 4 bars or more.

6.3 Discussion

Let us look closer at our results. As for the correctness ratio of theme classification in table 3, we see the more bars cause better results. In fact, in a case of 3 tones by Viterbi algorithm, we have improved the results of 95.7% with 4 bars to 100.0% with 8 bars. Similarly in a case of 4 tones by Forward algorithm, we have improved 69.6% with 4 bars to 82.6% with 8 bars, and 56.7% with 4 bars by Viterbi algorithm to 78.3% with 8 bars. However, in a case of 5 tones, we got the worse result of 39.1% with 4 bars by Viterbi algorithm to 30.4% with 8 bars. One of the reasons is that, the more bars we have, the more chances we get to make mistakes in a case of two similar themes. We have already pointed out this problem in another work [15].

Table 3. Correctness Ratio - (a) Theme and (b) Chord

	Algorithm	
(tone) bars	Chord(Forward)	(Viterbi)
(3) 4	100.0%	95.7%
(3) 8	100.0%	100.0%
(4) 4	69.6%	56.5%
(4) 8	82.6%	78.3%
(5) 4	39.1%	47.8%
(5) 8	30.4%	47.8%

	Number of Correct Bars		
tone	8bars	$\geq$ 6bars	$\geq$ 4 bars
3	4.35%	39.1%	69.6%
4	0.0%	38.9%	55.6%
5	0.0%	45.5%	63.6%

In this experiment, we have compared Viterbi and Forward algorithms with each other. In cases of correctness ratio of theme classification with respect to 3 tones and 4 tones, Forward algorithm is superior to Viterbi, while in a case of 5 tones, Viterbi is better. One of the specific points to Forward algorithm is that, the more bars we give, the worse the classification goes, but not in a case of Viterbi algorithm. This comes from the difference of probability calculation, though we skip the detail of probability results. In Forward algorithm, we get the probability by summarizing the values along with all the paths to the state of interests. On the other hand, in a case of Viterbi, we get the probability by finding one of the paths with the highest probability.

There is no investigation of polyphony classification to compare directly with our results. In [9,10], given about 3000 music of polyphony, classification as been considered as query and the results have been evaluated based precision. They got 59.0% at best.

Let us examine our previous results[14]. Naive Bayesian provides us with 87.0 % correctness ratio, and we got 91.3 % correctness ration at best by EM algorithm. Compared to our case where all the melodies are polyphony that are much complicated, we got the perfect correctness ratio (100.0 %) with 8 bars. Certainly our approach is promising.

As for chord estimation, we got up to 70% correctness ratio with respect to 4 bars and more cases. Basically it is possible to say that HMM approach works well.

7 Conclusion

In this work, we have proposed a sophisticated approach to classify polyphonic melodies given a small amount of training polyphonic music. To do that, we have introduced special features of pitch spectrum and estimated Maximum Likelihood based on HMM. We have shown the perfect estimation of theme classification by examining small amount of tones and bars in unknown melodies.

References

1. Dowling, W.J.: Scale and Contour – two components of a theory of memory for melodies. Psychological Reviews 85(4), 341–354 (1978)
2. Droettboom, M., et al.: An Approach Towards A Polyphonic Music Retrieval System. In: Intn'l Symp. on Music Information Retrieval (ISMIR) (2001)

3. Droettboom, M., et al.: Expressive and Efficient Retrieval of Symbolic Musical Data. In: Intn'l Symp. on Music Information Retrieval (ISMIR) (2002)
4. Grossman, D., Frieder, O.: Information Retrieval – Algorithms and Heuristics. Kluwer Academic Press, Dordrecht (1998)
5. Ishigeta, M., et al.: Theory of Music, Ongaku-No-Tomo-Sha (in Japanese) (2001)
6. Iwasaki, M.: Foundation of Incomplete Data Analysis, Economicst Sha (in Japanese) (2002)
7. Kim, Y., et al.: Analysis of A Contour-based Representation for Melody. In: Intn'l Symp. on Music Information Retrieval (ISMIR) (2000)
8. Miura, T., Shioya, I.: Similarities among Melodies for Music Information Retrieval. In: CIKM. ACM Conf. on Information and Knowledge Management (2003)
9. Pickens, J.: A Comparison of Language Modelling and Probabilistic Text Information Retrieval Approaches to Monophonic Music Retrieval. In: ISMIR. Intn'l Symp. on Music Information Retrieval (2000)
10. Pickens, J., Crawford, T.: Harmonic Models for Polyphonic Music Retrieval. In: CIKM. ACM Conf. on Information and Knowledge Management (2002)
11. Uitdenbogerd, A.L., et al.: Manipulation of Music For Melody Matching. In: ACM MultiMedia Conf. (1998)
12. Uitdenbogerd, A.L., et al.: Music Ranking Techniques Evaluated. In: ISMIR. Intn'l Symp. on Music Information Retrieval (2000)
13. Walshaw, C.: abc Version 1.6, http://www.gre.ac.uk/~c.walshaw/abc2mtex/abc.txt
14. Yoshihara, Y., Miura, T.: Melody Classification Using EM Algorithm. In: COMPSAC. Computer Software and Applications Conference, pp. 204–210 (2005)
15. Yoshihara, Y., Miura, T.: Classifying Polyphonic Music Based on Markov Model. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 697–706. Springer, Heidelberg (2006)

Elastic Non-contiguous Sequence Pattern Detection for Data Stream Monitoring

Xinqiang Zuo, Yuanbing Zhou, and Chunhui Zhao

State Power Economic Research Institute, China
{zuoxinqiang,zhouyuanbing,zhaochunhui}@chinasperi.com.cn

Abstract. In recent years, there has been an increasing interest in the detection of non-contiguous sequence patterns in data streams. Existing works define a fixed temporal constraint between every pair of adjacent elements of the sequence. While this method is simple and intuitive, it suffers from the following shortcomings: 1)It is difficult for the users who are not domain experts to specify such complex temporal constraints properly; 2)The fixed temporal constraint is not flexible to capture interested patterns hidden in long sequences. In this paper, we introduce a novel type of non-contiguous sequence pattern, named **Elastic Temporal Constrained Non-contiguous Sequence Pattern(ETC-NSP)**. Such a pattern defines an elastic temporal constraint on the sequence, thus is more flexible and effective as opposed to the fixed temporal constraints. Detection of ETC-NSP in data streams is a non-trivial task since a brute force approach is exponential in time. Our method exploits an similarity measurement called **Minimal Variance Matching** as the basic matching mechanism. To further speed up the monitoring process, we develop pruning strategies which make it practical to use ETC-NSP in streaming environment. Experimental studies show that the proposed method is efficient and effective in detecting non-contiguous sequence patterns from data streams.

1 Introduction

Sequence patterns, which may describe a meaningful tendency or an important phenomenon of the monitored objects, refer to a series of ordered elements. Sequence pattern detection in streaming environment is to find the data streams that contain the specified sequence patterns through subsequence matching. Given a distance function, a pattern is said to be contained in a data stream if the distance between the pattern and a subsequence of the data stream is within a predefined threshold.

Recently, there is an increasing interest in the monitoring of non-contiguous sequence patterns [2,3], which can be obtained from a long sequence by discarding some irrelevant parts. Non-contiguous sequence pattern is of great importance since some applications require retrieving a specific ordering of events without

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 599–608, 2007.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
S_1	4	x	x	x	7	x	x	x	x	5
S_2	x	4	x	7	x	x	5	x	x	x
S_3	4	x	x	x	x	x	7	x	5	x

Fig. 1. Fixed Temporal Constraint on Non-contiguous Sequence Patterns

caring about the events which interleave them in the actual sequence[2,3,5]. As in the above example, previous work related to non-contiguous sequence patterns define a fixed temporal constraint on the adjacent elements of a sequence, i.e, every pair of adjacent elements in the sequence should satisfy a temporal interval. This mechanism is convenient for indexing and searching for interested sequence patterns, but suffers from the following shortcomings: 1)It is difficult, if not impossible, for users who are not domain experts to define such non-contiguous patterns properly. 2)Though non-contiguous sequence patterns may repeat themselves in a long sequence, it is not likely that they always follow the fixed temporal constraint in a strict way. Thus the fixed temporal constraint is not flexible to capture interested patterns hidden in a long sequence. Consider the situation in Figure.1, where a non-contiguous sequence S_n=$(4, 7, 5)$ appears in 3 long sequences $\{S_1, S_2, S_3\}$ with different temporal intervals among its elements. Any fixed temporal constrained sequence pattern can only detect the occurrence of S_n in at most one of long sequences and omit the other two. And at least 3 non-contiguous sequence patterns has to be defined to detect all the occurrences of S_n. Obviously this strategy is ineffective in handling large amounts of sequence patterns and a natural requirement of detecting all the occurrences of the same sequence with one single pattern emerges.

In this paper, we introduce a novel type of non-contiguous sequence pattern, what we name **Elastic Temporal Constrained Non-contiguous Sequence Pattern(ETC-NSP)**. Such a pattern defines an elastic temporal constraint on the sequence, i.e, only the first and the last element of the sequence should satisfy a specified time-interval. This mechanism is more flexible and effective as opposed to the fixed temporal constraints.

Non-contiguous sequence pattern detection is not a trivial task. For a long sequence of length m, the number of its n-length non-contiguous subsequences is as many as C_m^n. A brute-force approach is exponential in time complexity since all the non-contiguous subsequence should be considered.

Motivated by be the requirement of elastic temporal constraint and the following challenge mentioned above, we exploit **Minimal Variance Matching(MVM)**[4] as the basic similarity model for efficient non-contiguous sequence pattern detection.

The contribution of this paper can be summarized as: 1)We propose the concept of elastic temporal constrained non-contiguous sequence pattern (ETC-NSP) in stream monitoring, which is more accurate and robust. 2)We reduce

the computation complexity of ETC-NSP from $O(C_m^n)$ to $O(mn)$ by exploiting the Minimal Variance Matching(MVM). 3)To further speed up the pattern detection process, we develop 2 pruning methods for ETC-NSP to make it feasible to be applied in stream monitoring.

The remainder of this paper is as follows: Section 2 describes related work on stream monitoring in the past years. A formal definition of ETC-NSP together with the corresponding MVM algorithm is provided in Section 3. We propose the pruning methods to speed up ETC-NSP in Section 4. Section 5 describes the experimental evaluation of ETC-NSP on different data sets, which shows the efficiency and effectiveness of ETC-NSP.

2 Related Work

The problem of sequence pattern detection has been studied extensively in the past years. In [1], a solution for similar sequence matching has been proposed, where the trick of Discrete Fourier Transformation(DFT) has been used. After the features of sequence be extracted by DFT, they are indexed by R-Tree. The method of Minimal Variance Matching(MVM) is developed in [4]. MVM handles the matching of sequences of different lengths, but it is only applicable in static environment.

The first work about non-contiguous sequence is proposed in [3], but only fixed temporal constraint was considered. Another important work about non-contiguous sequence is [2]. Indexing methods are used in both [3] and [2] to speed up the calculation process. In [6], the authors solved the problem of non-contiguous sequence patterns in both spatial dimension and temporal dimension. The sequence pattern monitoring problem is well studied in [7] under the notation of Dynamic Time Warping(DTW) distance, where elastic temporal constraint is not considered, either.

3 Non-contiguous Subsequence

3.1 Problem Formalization

We first give a formal definition of the non-contiguous subsequence.

Definition 1. ***Non-contiguous Subsequence(NCS).*** *Given a long sequence of length m:* $X=(X_1, X_2, ..., X_m)$*, where* X_i *is the* i_{th} *element of* X*. A non-contiguous subsequence of* X *is a sequence* $X^{'}=(X_{i_1}, X_{i_2}, ..., X_{i_n})(n \leq m)$*, where* $1 \leq i_1 \leq i_2 \leq ... \leq i_n \leq m$*, denoted as* $X^{'} \subset X$

Definition 2. ***Fixed Temporal Constrained Non-Contiguous Sequence Pattern(FTC-NSP).*** *A fixed temporal constrained non-contiguous sequence pattern is a non-contiguous sequence with fixed temporal constraints.* $P =< (P_1, t_1), (P_2, t_2), ..., (P_n, t_n) >$*. A long sequence* S *contains a FTC-NSP* P*, if there exists a non-contiguous sequence* $S^{'} =< (S_1^{'}, t_1^{'}), (S_2^{'}, t_2^{'}), ..., (S_n^{'}, t_n^{'}) >, S^{'} \subset S$

such that $\sum_{i=1}^{n} |P_i - S_i^{'}| < \epsilon$ [1] *and* $|t_i - t_{i-1}| = |t_i^{'} - t_{i-1}^{'}|$, $\forall i \in 2, ..., n$, *where* ϵ *is a given threshold.*

From the above definition, it can be found that every pair of the adjacent elements in a non-contiguous sequence should satisfy the specified temporal constraint. Though a non-contiguous sequence pattern may repeat itself in data streams, it is not likely that the time interval between its elements may remain unchanged all the time. Thus the fixed temporal constraint is not flexible and effective enough to capture interested patterns hidden in data streams.

Definition 3. ***Elastic Temporal Constrained Non-contiguous Sequence Pattern(ETC-NSP).*** *An elastic temporal constrained non-contiguous sequence pattern is a non-contiguous sequence* $P=(P_1, P_2, ..., P_n)$ *together with a temporal interval* t_p. *A long sequence* S *contains a ETC-NSP* P *if there exists a non-contiguous sequence* $S^{'} =< (S_1^{'}, t_1^{'}), (S_2^{'}, t_2^{'}), ..., (S_n^{'}, t_n^{'}) >, S^{'} \subset S$ *such that* $\sum_{i=1}^{n} |P_i - S_i^{'}| < \epsilon$ *and* $t_n^{'} - t_1^{'} \leq t_p$, *where* ϵ *is a given threshold.*

In ETC-NSP, the temporal constraint is more flexible since only the first and last element of the sequence should satisfy the temporal interval. Thus it is tolerant to time shift in the sequence and robust to noise, and is more powerful in capturing hidden patterns. Consider the example in Figure.1, the pattern in the 3 sequence can be detected by a ETC-NSP $(4, 7, 5)$ with a temporal interval 10.

3.2 ETC-NSP Detection in Streaming Environment

In this subsection, we discuss the problem of ETC-NSP detection in data streams. A data stream is a non-contiguous, ordered sequence of values $x_1, x_2, ..., x_n, ...$, where x_n is the most recent values[7]. Due to the semi-infinite characteristic of data stream, it is not feasible to monitor on the whole stream. We exploit a sliding window in the streaming environment, and search for non-contiguous subsequence in the window. We now define the distance between an ETC-NSP and a data stream.

Definition 4. ***Distance Between ETC-NSP and Data Stream.*** *Given a data stream* S *and an ETC-NSP* P *with a temporal constraint* t_p. *We apply a sliding window* W *of length* $m=|P|+t_p$, *where* $|P|$ *is the length of* P, *on the data stream.* $W^{'}$ *is a non-contiguous subsequence of length* $|P|$ *in* W, *and* $\mathcal{W}$ *is the set of all the non-contiguous subsequence of length* $|P|$ *in* W. *The distance between* S *and* P *can be defined*

$$D(S, P) = \min\{\mathcal{D}_{L_1}(P, W^{'}), W^{'} \in \mathcal{W}\}$$

From the above definition, we can see that the distance between a sequence pattern and a data stream is the minimal value of the L_p distances between the sequence pattern and all the subsequences in a sliding window applied on thedata

[1] Note that any L_p distance can be used here. Without the loss of generality, we exploit the L_1 distance.

stream. If the pattern sequence is seemed to be a query, pattern detection in data stream can be considered as a range query process.

Definition 5. ***Non-Contiguous Subsequence Based Range Query.*** *Given a data stream S, an ETC-NSP P with a temporal constraint t_p and a threshold ϵ. When applying a sliding window W of length m=$|P|$+t_p on the data stream, the range query process is to determine whether the distance between S and P is within ϵ, i.e., $D(S, P) < \epsilon$*

4 Proposed Method

4.1 Naive Approach

It can be found from Definition.4 that the distance between a sequence pattern and a data stream is in essence the distance between the sequence pattern and the sliding window applied on the data stream, which is in the form of a long sequence. Given a long sequence X of length m and a pattern sequence P of length $n(n \leq m)$, the distance between X and P is the minimum L_p distance between P and all the non-contiguous subsequences of X.

Lemma 1. *For a data sequence X of length m, there are C_m^n non-contiguous subsequences of X whose lengths are $n(n \leq m)$.*

A brute-force approach to calculate $D(X, P)$ is to enumerate all the non-contiguous subsequences of X, for each subsequence $X^{'}$, calculate the L_P distance between $X^{'}$ and P. The minimum $D_{L_p}(X^{'}, P)$ is the distance between $X^{'}$ and P. Without loss of generality, we exploit L_1 distance in the rest of this paper. This simple approach is rather straightforward and understandable, but is precluded due to its extremely expensive time cost.

4.2 Minimal Variance Matching

Since the naive method to calculate the distance is rather expensive in terms of time cost, we exploit a more efficient technique, the **Minimal Variance Matching(MVM)**, which outperforms the naive method in a magnitude of time (from $O(n * C_m^n)$ to $O(m * (m - n))$. MVM is first proposed in [4] and considered as a directed acyclic graph problem. In this paper, we formulate this technique in a more formal dynamic programming manner.

We first introduce the notion of non-contiguous path. Let M be a $m \times n$ matrix of pairwise distances between elements of X and P, i.e., $M[i][j]$=$|X_i - P_j|$. A non-contiguous path $P_{nc} = (m_1, m_2, ..., m_n)$ is a sequence of n matrix cells, $m_k = M[i_k][k], k \leq i_k \leq (k + m - n)$ and $i_k < i_{k+1}$. Any non-contiguous path determines an alignment between X and P, thus there are total C_m^n different non-contiguous paths for X and P. The distance between X and P corresponds to the non-contiguous path that has the minimum sum of matrix cells, i.e., $D(X, P) = \min_{P_{nc}}\{\sum_{i=1}^{n} m_i, m_i \in P_{nc}\}$. The algorithm of finding such a non-contiguous path is described in Algorithm.1.

Algorithm 1. Calculate non-contiguous subsequence distance

Input: Long sequence $X = (X_1, ..., X_m)$ and pattern sequence $P = (P_1, ..., P_n), n \leq m$
Output:non-contiguous subsequence distance between X and P

```
1: for j = 1 to n do
2:     for i = j to j + m − n do
3:         M[i][j]=|X_i − Y_j|
4:     end for
5: end for
6: for j = 2 to n do
7:     for i = j to j + m − n do
8:         M[i][j]+=min_{j−1≤k≤i−1} {M[k][j − 1]}
9:     end for
10: end for
11: return min_{j≤k≤j+m−n} {M[k][j]}
```

Now we provide a running example in Figure.2 to illustrate the method in more detail. Consider the two sequences X=$(5, 3, 1, 19, 4, 3, 2)$ and Y=$(4, 3, 6, 1, 2)$, we show how to obtain $D(X, Y)$ with our proposed algorithm. The left table in Figure.2 shows the initial state of the matrix M. In the right table, the series of cells with shadow is corresponding to the non-contiguous path and the minimal value 5 in the right-most column is the MVM distance between X and Y. That is to say, among all the subsequences of X, $X^{'}$=$(5, 3, 4, 3, 2)$ has the minimal L_1 distance 5 with Y. This algorithm is based on a dynamic programming technique and has a time complexity of $O(m*(m-n+1))$, which is a great improvement as opposed to the naive method. One important property of the MVM distance we will use later is that if the two sequences being calculated are of the same lengths, then MVM distance is identical to L_1 distance.

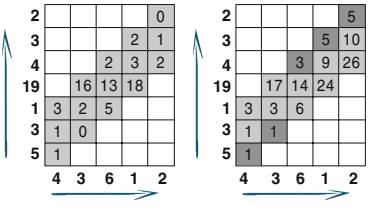


Fig. 2. Example of the MVM Algorithm

5 Pruning Methods in Streaming Environment

Although the method in last section already outperforms the naive approach by orders of magnitude, a time complexity of $O(m*(m-n+1))$ is still needed. Thus it is still impractical in streaming environment where data elements are coming continuously at high rate. We propose two techniques in this section to further speed up the monitoring process and make it feasible to be used in streaming environment.

5.1 Triangle Inequality

In metric space, there exists a triangle inequality which is used as a pruning tool in the search process, that is: $D_{L_p}(X,Y) + D_{L_p}(Y,Z) > D_{L_p}(X,Z)$. In this subsection, we prove that the MVM distance also follows a triangle inequality which is different from that in the metric space.

Theorem 1. *triangle Inequality Theorem: Given three data sequence Q, S and R. Q is a long sequence of length m. S and R are short sequences of length n. The following inequality holds: $D(Q,S) + D(S,R) > D(Q,R)$.*

5.2 Lower Bound Distance

In this subsection we propose a lower bound distance of the MVM distance [8]. Suppose that P=$(P_1, P_2, ..., P_n)$ is an ETC-NSP with a temporal constraint t_p and a threshold ϵ, and W=$(W_1, W_2, ..., W_n, ..., W_{n+t_p})$ is a sliding window on a data stream S. We define the data structure W^L=$(W_1^L, W_2^L, ..., W_n^L)$ and W^U=$(W_1^U, W_2^U, ..., W_n^U)$, where

$$W_i^L = \min\{W_i, .., W_{i+t_p}\}, W_i^U = \max\{W_i, .., W_{i+t_p}\}$$

We can derive a lower bound distance $D_{lb}(P,W)$ of the MVM distance between W and P from the proposed W^L and W^U,

$$D_{lb}(P,W) = \sum_{i=1}^{n} g(P_i, W_i^L, W_i^U)$$

$$g(P_i, W_i^L, W_i^U) = \begin{cases} |P_i - W_i^L| & \text{if } P_i \le W_i^L \\ |P_i - W_i^U| & \text{if } P_i \ge W_i^U \\ 0 & \text{otherwise} \end{cases}$$

The proposed lower bound distance $D_{lb}(P,W)$ can be calculated in a linear time and used as a pruning tool. When $D_{lb}(P,W) > \epsilon$, it is straightforward that $D(P,W) > \epsilon$ since $D(P,W) > D_{lb}(P,W)$. Then it is unnecessary to calculate $D(P,W)$ explicitly.

5.3 Complexity Analysis

Compared with the naive method, the novel algorithm uses a matrix to reach the lower bound distance. Given m the length of the query data and n the length of the incomplete data, the time complexity of this method is $\mathcal{O}(n(m-n+1))$. As can be found directly from the running example, for each of the n columns in the matrix, $m-n+1$ cells has to be visited in our algorithm. Thus this method is especially efficient when m and n are close. One extreme case is when m is the same as n, our algorithm is identical to the L_1 distance and the time complexity has been reduced to $\mathcal{O}(m)$. That is, since no value is missing, the lower bound of the L_1 distance is just itself. Another extreme case is when n is much smaller than m, i.e., a large amount of elements is missing, the time complexity is as large as $O(mn)$. The two pruning methods further reduce the time complexity since it is not necessary to really conduct MVM calculation for all the data.

6 Experiment Evaluation

To evaluate the effectiveness of our method, we carried out experiments on both real and synthetic data sets. Our experiments were conducted on an Intel CPU of 1.7GHz with 512 MB of Memory. The programming language we exploited was Java(JDK 1.5).

6.1 Data Set

- Synthetic Data Set: The data set consists of a synthetic data stream, which contains 4000 predefined non-contiguous sequence patterns interleaved by white noise. The noise and the values of the sequence patterns follow the same Gaussian distributions.
- S&P 500 Data Set: This data contains stock information about 500 companies for about 250 days, including the prices of all the 500 stocks in the form of time series.[2]

6.2 Capture of Non-contiguous Sequence Patterns

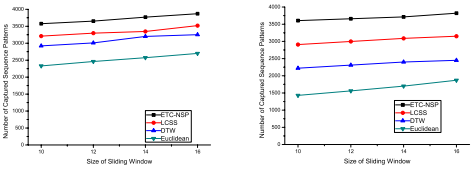
We present the power of our method in detecting of non-contiguous sequence patterns in data streams in this subsection. We generate two data sets with different Gaussian distributions, i.e, with $< mean = 10, variance = 2 >$ and $< mean = 10, variance = 5 >$ respectively. in Figure.3, we compare the number of sequence patterns captured by different similarity measurements, including LCSS, DTW, Euclidean distance, and the proposed ETC-NSP based MVM. For LCSS, it is needed that a parameter has to be specified to judge whether two real-valued data elements are "equal". The perform of LCSS will vary according to this parameter. We show the best performance of LCSS in the figures. ETC-NSP outperforms other technique in terms of number of sequence patterns captured. When the variance of the distribution decreases, the performance of LCSS, DTW and Euclidean degrades, but ETC-NSP is influenced slightly by this change. This shows that ETC-NSP is robust to noise when applied to monitor sequence patterns. The elastic temporal constraint allows it to avoid the noise element in the data stream automatically. DTW and Euclidean distance are sensitive to noise, so they dismiss many sequences in the presence of noise.

6.3 Pruning Power

The proposed triangle inequality and the lower bound can be used to pruning unqualified data sequences in the monitoring process. The measurement for its effect is pruning power, which can be defined as follows:

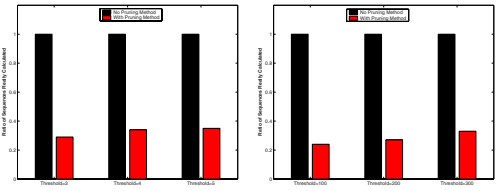
$$P = \frac{NumberofObjectsDoNotRequireFullComputation}{NumberofObjectsInDataset}$$

[2] http://biz.swcp.com/stocks/



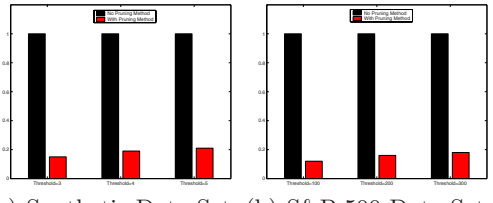
(a) Mean=10,Variance=2 (b) Mean=10,Variance=5

Fig. 3. Power of Pattern Detection



(a) Synthetic Data Set (b) S&P 500 Data Set

Fig. 4. Effectiveness of Pruning with Lower Bound



(a) Synthetic Data Set (b) S&P 500 Data Set

Fig. 5. Effectiveness of Pruning with triangle Inequality

We combine the lower bound and the triangle inequality to prune unqualified sequence data in the monitoring process. The pruning power on both the data sets is shown in Figure.4 and Figure.5. When the threshold to judge whether a sequence is a defined pattern increases, the pruning power of method decrease. This is because that if the calculated lower bound is not larger than the threshold, it is not permitted to dismiss the sequence data without conduct the MVM algorithm.

7 Conclusion

We propose the problem of elastic temporal constrained non-contiguous subsequence detection in data streams. The MVM distance is used to evaluate the

similarity between sequence patters and data in streaming environment. To further speed up MVM, we develop two pruning methods, i.e, the triangle inequality and the lower bound to make it feasible in streaming environment.

References

1. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast Subsequence Matching in Time-Series Databases. In: Proceedings 1994 ACM SIGMOD Conference, Mineapolis, MN (1994)
2. Mamoulis, N., Yiu, M.L.: Non-contiguous Sequence Pattern Queries. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K., Ferrari, E. (eds.) EDBT 2004. LNCS, vol. 2992, Springer, Heidelberg (2004)
3. Wang, H., Perng, C.-S., Fan, W., Park, S., Yu, P.S.: Indexing Weighted-Sequences in Large Databases. In: Proceedings of 19th International Conference On Data Engineering (2003)
4. Latecki, L.J., Megalooikonomou, V., Wang, Q., Lakaemper, R., Ratanamahatana, C.A., Keogh, E.: Elastic Partial Matching of Time Series. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, Springer, Heidelberg (2005)
5. Hadjieleftheriou, M., Kollios, G., Bakalov, P., Tsotras, V.J.: Complex Spatio-Temporal Pattern Queries. In: Proceedings of the 31st International Conference on Very Large Data Bases (2005)
6. Hadjieleftheriou, M., Kollios, G., Tsotras, V.J., Gunopulos, D.: Efficient Indexing of Spatiotemporal Objects. In: Jensen, C.S., Jeffery, K.G., Pokorný, J., Šaltenis, S., Bertino, E., Böhm, K., Jarke, M. (eds.) EDBT 2002. LNCS, vol. 2287, Springer, Heidelberg (2002)
7. Sakurai, Y., Faloutsos, C., Yamamuro, M.: Stream Monitoring under the Time Warping Distance. In: International Conference on Data Engineering(ICDE) (2007)
8. Keogh, E.J.: Exact indexing of dynamic time warping. In: VLDB 2002 (2002)

Joint Cutoff Probabilistic Estimation Using Simulation: A Mailing Campaign Application*

Antonio Bella, Cèsar Ferri, José Hernández-Orallo, and María José Ramírez-Quintana

Universitat Politécnica de Valencia, DSIC, Valencia, Spain

Abstract. Frequently, organisations have to face complex situations where decision making is difficult. In these scenarios, several related decisions must be made at a time, which are also bounded by constraints (e.g. inventory/stock limitations, costs, limited resources, time schedules, etc). In this paper, we present a new method to make a good global decision when we have such a complex environment with several local interwoven data mining models. In these situations, the best local cutoff for each model is not usually the best cutoff in global terms. We use simulation with Petri nets to obtain better cutoffs for the data mining models. We apply our approach to a frequent problem in customer relationship management (CRM), more specifically, a direct-marketing campaign design where several alternative products have to be offered to the same house list of customers and with usual inventory limitations. We experimentally compare two different methods to obtain the cutoff for the models (one based on merging the prospective customer lists and using the local cutoffs, and the other based on simulation), illustrating that methods which use simulation to adjust model cutoff obtain better results than a more classical analytical method.

1 Introduction

Data mining is becoming more and more useful and popular for decision making. Single decisions can be assisted by data mining models, which are previously learned from data. Data records previous decisions proved good or bad either by an expert or with time. This is the general picture for predictive data mining. The effort (both in research and industry) is then focussed on obtaining the best possible model given the data and the target task. In the end, if the model is accurate, the decisions based on the model will be accurate as well.

However, in real situations, organisations and individuals must make several decisions for several given problems. Frequently, these decisions/problems are interwoven with the rest, have to be made in a short period of time, and are accompanied with a series of constraints which are also just an estimation of the

* This work has been partially supported by the EU (FEDER) and the Spanish MEC under grant TIN 2007-68093-C02-02, Generalitat Valenciana under grant GV06/301, UPV under grant TAMAT and the Spanish project "Agreement Technologies" (Consolider Ingenio CSD2007-00022).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 609–619, 2007.

real constraints. In this typical scenario, making the best local decision for every problem does not give the best global result. This is well-known in engineering and decision making, but only recently acknowledged in data mining. Examples can be found everywhere: we cannot assign the best surgeon to each operation in a hospital, we cannot keep a fruit cargo until their optimal consumption point altogether, we cannot assign the best delivering date for each supplier, or we cannot use the best players for three matches in the same week.

In this context, some recent works have tried to find optimal global solutions where the local solutions given by local models are not good. These works address specific situations: rank aggregation [3] and cost-sensitive learning are examples of this, a more general "utility-based data mining"[1] also addresses this issue, but also some other new data mining tasks, such as quantification [5], are in this line. Data mining applied to CRM (Customer-Relationship Management) [1] is also one of the areas where several efforts have also been done.

Although all these approaches can be of great help in specific situations, most of the scenarios we face in real data mining applications do not fit many of the assumptions or settings of these previous works. In fact, many real scenarios are so complex that the "optimal" decision cannot be found analytically. Approximate, heuristic or simplified global models must be used instead. One appropriate non-analytic way to find good solutions to complex problems where many decisions have to be made is through simulation.

In this work, we connect inputs and outputs of several data mining models and simulate the global outcome under different possibilities. Through the power of repeating simulations after simulations, we can gauge a global cutoff point in order to make better decisions for the global profit. It is important to highlight that this approach does not need that local models take the constraints into account during training (i.e. models can be trained and tested as usual). Additionally, we can use data which has been gathered independently for training each model. The only (mild) condition is that model predictions must be accompanied by probabilities (see e.g. [4]) or certainty values, something that almost any family of data mining algorithms can provide. Finally, probabilities and constraints will be used at the simulation stage for estimating the cutoff.

In order to do this, we use the basic Petri Nets formalism [6], with additional data structures, as a simple (but powerful) simulation framework and we use probabilistic estimation trees (classical decision trees accompanied with probabilities [4]). We illustrate this with a very frequent problem in CRM: we apply our approach to a direct-marketing campaign design where several alternative products have to be offered to the same house list of customers. The scenario is accompanied, as usual, by inventory/stock limitations. Even though this problem seems simple at the first sight, there is no simple good analytic solution for it. In fact, we will see that a reasonable analytic approach to set different cutoffs for each product leads to suboptimal overall profits. In contrast, using a joint cutoff probabilistic estimation, which can be obtained through simulation, we get better results.

[1] (http://storm.cis.fordham.edu/~gweiss/ubdm-kdd05.html)

The paper is organised as follows. Section 2 sets the problem framework, some notation and illustrates the analytical (classical) approach. Section 3 addresses the problem with more than one product and presents two methods to solve it. Section 4 includes some experiments with the presented methods. The paper finishes in Section 5 with the conclusions.

2 Campaign Design with One Product

Traditionally, data mining has been widely applied to improve the design of mailing campaigns in Customer Relationship Management (CRM). The idea is simple: discover the most promising customers using data mining techniques, and in this way, increase the benefits of a selling campaign.

The process begins by randomly selecting a sample of customers from the company database (house list). Next, all these customers receive an advertisement of the target product. After a reasonable time, a minable view is constructed with all these customers. In this table, every row represents a different customer and the columns contain information about customers; the predictive attribute (the target class) is a Boolean value that informs whether the corresponding customer has purchased or not the target product. Using this view as a training set, a probability estimation model is learned (for instance a probability estimation tree). This model is then used to rank the rest of customers of the database according to the probability of buying the target product. The last step is to select the optimal cutoff that maximises the overall benefits of the campaign, i.e. the best cutoff of the customer list ranked by customer buying probability.

The optimal cutoff can be computed using some additional information about some associated costs: the promotion material cost (edition costs and sending cost)($Icost$), the benefit from selling one product (b) and the cost to send an advertisement to a customer ($cost$). Given all this information, the *accumulated expected benefit* for a set of customers is computed as follows. Given a list C of customers, sorted by the expected benefit (for $c_k \in C, E\_benefit(c_k) = b \times p(c_k) - cost$), we calculate the *accumulated expected benefit* as $-Icost + \sum_{k=1}^{j} b \times p(c_k) - cost$, where $p(c_k)$ is the estimated probability that customer c_k buys the product and j is the size of the sample of customers to which a pre-campaign has offered the product. The optimal cutoff is determined by the value $k, 1 \leq k \leq j$ for which the greatest accumulated expected benefit is obtained.

The concordance between the real benefits with respect to the expected benefits is very dependent on the quality of the probability estimations of the model. Therefore, it is extremely important to train models that estimate accurate probabilities (e.g. see [4]). A more reliable estimation of the cutoff can be obtained by employing different datasets of customers (or by spliting the existing dataset): a training dataset for learning the probability estimation models, and a validation dataset to compute the optimal cutoff. With this validation dataset the latter estimation of the *accumulated expected benefit* turns into a real calculation of the *accumulated benefit*, where $p(c_k)$ is changed by $f(c_k)$ in the formula, being $f(c_k)$ the response of c_k wrt. the product, such that $f(c_k) = 0$ if customer c_k does not

Table 1. Accumulated expected benefit vs. Accumulated benefit

Customer	Buys	Probability	E(Benefit)	Acc. Exp. Benefit	Acc. Benefit
				-250	-250
3	YES	0.8098	141.96	-108.04	-70
10	YES	0.7963	139.26	31.22	110
8	YES	0.6605	112.10	143.31	290
1	YES	0.6299	105.98	249.30	470
4	NO	0.5743	94.86	344.15	450
6	NO	0.5343	86.85	431.00	430
5	YES	0.4497	69.94	500.94	**610**
7	NO	0.2675	33.50	534.44	590
9	NO	0.2262	25.24	**559.68**	570
2	NO	0.0786	-4.29	555.39	550

buy the product and $f(c_k) = 1$ if c_k buys it. Then, the cutoff is determined by the greatest accumulated benefit.

Let us see an example where the benefit for the product is 200 monetary units (m.u.), the sending cost is 20 m.u. and the investment cost is 250 m.u. In Table 1 we compare the results obtained with each method. According to the *accumulated expected benefit* we will set the cutoff at 90% of the customers, which clearly differs from the maximum *accumulated benefit* (located at 70%).

3 Using Simulation and Data Mining for a Campaign Design with More Than One Product

The approach shown at the previous section has been successfully applied to very simple cases (i.e. one single product for each campaign), but computing optimal cutoffs by analytic methods is impossible for more complex scenarios (more than one product, constraints for the products, etc.). Therefore, in this section we develop two different approaches: one is an extension of the analytic method, and the other is a more novel and original method based on simulation.

Back on our marketing problem, the objective now is to design a mailing campaign offering N products to a customer list, but taking the following constraints into consideration: there are stock limits (as usual), each product has a different benefit, and the products are alternative, which means that each customer would only buy one of them (or none). As we have seen at Section 2, a good solution, at least apriori, could be to determine a cutoff point defining the segment of customers we have to focus on. But now, since there are several products, it is not clear how this cutoff can be defined/determined. Based on the idea of sorting the customers by their expected benefit, one possibility (what we call the *single approach*) is to combine (in some way, like adding or averaging) the optimal cutoffs which are analytically calculated for each product, in order to obtain a unique cutoff for the global problem. An alternative method, that we call *joint simulation approach*, is to determine in a dynamic way the global cutoff. We use

a validation set to simulate what will happen in a real situation if the customer receives the advertisement (of any of the N products).

Considering that all products have the same sending cost (*cost*), we define the following two alternative ways for obtaining a global cutoff using a validation set C:

1. **Single Approach:** For each product i, we downwardly sort C by the expected benefit of the customers, obtaining N ordered validation sets C_i (one for each product i). Now, for each C_i, $1 \leq i \leq N$, we determine its local cutoff point as we have explained in Section 2. Then, the global cutoff T is obtained by averaging the local cutoffs. In order to apply it, we now jointly sort the customers by their expected benefit considering all products at the same time (that is, just one ranked list obtained by merging the sets C_i). That produces as a result a single list SC where each customer appears N times. Finally, the cutoff T is applied over SC. Then, the real benefit obtained by this method will be the *accumulated benefit* for the segment of customers that will receive the advertisement for the total house list, which will be determined by this cutoff T.
2. **Joint Simulation Approach:** Here, from the beginning, we jointly sort the customers downwarded by their expected benefit of all the products, i.e. we merge the N sets C_i. However, we do not use local cutoffs to derive the global cutoff, but we calculate the cutoff by simulating $N \times |C|$ *accumulated benefits* considering all the possible cutoffs $T_j, 1 \leq j \leq N \times |C|$, where T_1 is the cutoff that only considers the first element of SC, T_2 is the cutoff that considers the two first elements of SC, and so on. Then, the best *accumulated benefit* gives the global cutoff.

To illustrate these two approaches consider a simple example consisting of 10 customers, 2 products (p_1 and p_2) and the parameters $Icost_{p_1} = 150$, $Icost_{p_2} = 250$, $b_1 = 100$, $b_2 = 200$, and $cost = 20$. Table 2 Left shows for each product the list of customers sorted by its expected benefit as well as the local cutoffs marked as horizontal lines. As we can observe, the cutoffs for products p_1 and p_2 are 90% and 70% respectively. Table 2 Right shows the global set and the global cutoff, which is marked by an horizontal line, computed by each approach. Note that the cutoff computed by the single and joint simulation methods is different. For the *single approach*, the global cutoff is 80% (the average of 90% and 70%), whereas the cutoff computed by the *joint simulation approach* is 90%.

We have adopted Petri nets [6] as the framework to formalise the simulation. Petri nets are well-known, easy to understand, and flexible. Nonetheless, it is important to highlight that the method we propose can be implemented with any other discrete simulation formalism. We used a unique Petri net to simulate the behaviour of all the customers, but we also implemented additional data structures to maintain information about customers and products (e.g. remaining stock for each product, remaining purchases for each customer). The Petri net can work with as many products and customers as we need with no change in the Petri net structure. Other similar problems, as mailing campaigns with

Table 2. Left: Customers sorted by their expected benefit for the case of two products. Right: Customers and cutoff for the Single and Joint Simulation Approaches.

Product p_1			
Customer	**E(Benefit)**	f_{p_1}	**Acc. Benefit**
			-150
2	76.61	1	-70
8	75.71	1	10
9	60.37	0	-10
5	48.19	1	70
1	44.96	1	150
7	30.96	0	130
10	24.58	1	210
3	23.04	0	190
6	7.81	1	270
4	-4.36	0	250

Product p_2			
Customer	**E(Benefit)**	f_{p_2}	**Acc. Benefit**
			-250
3	141.96	1	-70
10	139.26	1	110
8	112.10	1	290
1	105.98	1	470
4	94.86	0	450
6	86.85	0	430
5	69.94	1	610
7	33.50	0	590
9	25.24	0	570
2	-4.29	0	550

	Single & Joint Approaches			
	Customer	**Product**	**Acc. Benefit**	
			-400	
	3	p_2	-220	
	10	p_2	-40	
	8	p_2	140	
	1	p_2	320	
	4	p_2	300	
	6	p_2	280	
	2	p_1	360	
	8	p_1	440	
	5	p_2	620	
	9	p_1	600	
	5	p_1	680	
	1	p_1	760	
	7	p_2	740	
	7	p_1	720	
	9	p_2	700	
Single	10	p_1	780	
	3	p_1	760	
	6	p_1	840	Joint
	2	p_2	820	
	4	p_1	800	

non-alternative products, can also be handled without changes. Figure 1 shows our Petri net which has 24 places and 10 transitions. Each customer arrives to the Petri net and, thanks to the additional data structures created, the suitable number of tokens are put in each place to allow for the suitable transitions to be enabled/disabled and fired or not. E.g. if the remaining stock of the product is not zero a place *P12* is updated with as many tokens as the current stock is, and place *P11* is put to zero. The first place enables the transition *T4* that can be fired if the rest of conditions are fulfilled (place *P14* has a token), while the second place disables the transition *T5* that cannot be fired. Only two arcs have a weight not equal to one, the arc with the *product benefit* and the arc with the *sending cost*. The first arc finishes in the place *P1* (*Total gross benefit*) and the second one finishes in the place *P15* (*Total loss*). The total (or net) benefit for each cutoff is calculated subtracting the number of tokens accumulated in the places *P1* and *P15* (that is, *Total gross benefit* - *Total loss*).

In this scenario, we consider that, at the most, only one of the N products can be bought since they are alternative products (e.g. several cars or several houses or different brands for the same product). This constraint suggests to offer to

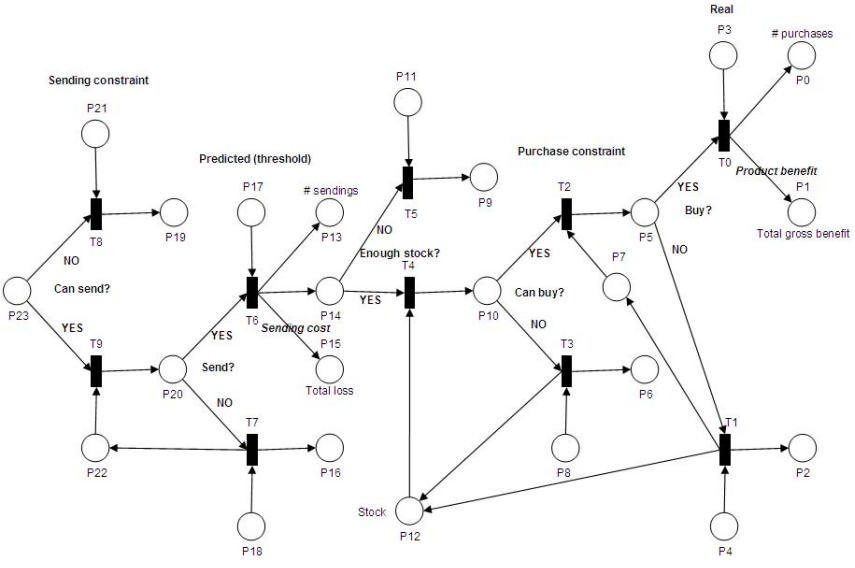


Fig. 1. Petri net for our mailing campaign

each customer only the product with the higher probability of being bought. If we impose this condition then we say that the approach is *with discarding*. In an approach with discarding, only the first appearance of each customer is taken into account. For instance, in the *single approach*, only the first occurrence of each customer in the customer segment determined by the global cutoff is preserved. Analogously, in the *joint simulation approach*, the simulation process does not consider customers that have been already processed. However, since a prospective customer who receives an offer might finally not buy the product, we consider an alternative option which allows several offers to the same customer. This approach is called *without discarding*. The combination of the two approaches and the two options for considering customer repetitions give four scenarios that will be experimentally analysed in the following section. The notation used for referring to these four different methods is: *Single WO* (Single approach without discarding), *Single WI* (Single approach with discarding), *Joint WO* (Joint simulation approach without discarding), and *Joint WI* (Joint simulation approach with discarding).

4 Experiments with N Products

For the experimental evaluation, we have implemented the four methods explained at Section 3 and the Petri net in Java, and have used machine learning algorithms implemented in the data mining suite WEKA [7].

4.1 Experimental Settings

For the experiments we have taken a customers file (*newcustomersN.db*) from the SPSS Clementine[2] samples, as a reference. This file has information about only 200 customers, with 8 attributes for each one, 6 of them are nominal and the rest are numeric. The nominal attributes are the *sex* of the customers (male or female), *region* where they live (inner city, rural, town, suburban), whether they are *married*, whether they have *children*, whether they have a *car* and whether they have a *mortgage*. The numeric attributes are the *age* of the customers and their annual *income*.

Since 200 customers are too few for a realistic scenario, we have implemented a random generator of customers. It creates customers keeping the attribute distributions of the example file, i.e. for numeric attributes it generates a random number following a normal distribution with the same mean and deviation as in the example file, and for nominal attributes it generates a random number keeping the original frequency for each value of the attributes in the file.

Also, to assign a class for each customer (wether s/he buys the product or not), we implemented a model generator. This model generator is based on a random decision tree generator, using the attributes and values randomly to construct the different levels of the tree. We have two parameters which gauge the average depth of the tree and most importantly, the probability of buying each product. We will use these latter parameters in the experiments below.

So, the full process to generate a customer file for our experiments consists of generating the customer data with our random generator of customers and to assign the suitable class with a model obtained by our model generator.

Finally, these are the parameters we will consider and their possible values:

- Number of customers: 10000 (60% training, 20% validation and 20% testing)
- Number of products: 2, 3 and 4
- Probability of buying each product: 0.01, 0.05, 0.2, 0.5, 0.8, 0.95 or 0.99
- Benefits for each product: 100 monetary units (m.u.) for the product 1 and 100, 200, 500 or 1000 m.u. for the other products
- Sending cost (the same for all products): 10, 20, 50 or 90 m.u.
- Stock for each product: 0.1, 0.2, 0.5 or 1 (multiplied by number of customers)
- Investment cost for each product: benefits of the product multiplied by stock of the product and divided by 20
- Correlation (how similar the products are): 0.25, 0.5, 0.75 or 1

4.2 Experimental Results

The three main experiments consist in testing 100 times the four approaches for 2, 3 and 4 products, where all the parameters are selected randomly for the cases where there are several possible values.

If we look at overall results, i.e. averaging all the 100 experiments, as shown in Table 3, the results for 2, 3 and 4 products are consistent. As suggested in [2]

[2] (http://www.spss.com/clementine/)

Table 3. Friedman test: wins (√) /loses (X)/draws(=)

	2 products				3 products				4 products			
	S.WO	**S.WI**	**J.WO**	**J.WI**	**S.WO**	**S.WI**	**J.WO**	**J.WI**	**S.WO**	**S.WI**	**J.WO**	**J.WI**
Benefits	165626	164568	171225	169485	182444	184077	186205	185694	220264	228483	231771	233724
S.WO	-	=	√	√	-	√	√	√	-	√	√	√
S.WI	=	-	=	√	X	-	√	=	X	-	=	√
J.WO	X	=	-	=	X	X	-	X	X	=	-	√
J.WI	X	X	=	-	X	=	√	-	X	X	X	-

we calculate a Friedman test and obtain that the four treatments do not have identical effects, so we calculate a post-hoc test (with a probability of 99.5%) This overall difference is clearly significant, as the significant analysis shown in Table 3, illustrates that the joint simulation approaches are better than the single ones. About the differences between with or without discarding methods, in the case of 2 products there are no significant differences. For 3 products the Single WI method wins the Single WO method, and the Joint WO method wins the Joint WI method. In the case of 4 products the approaches with discarding win the approaches without them. Moreover, in the case of 3 products, the Joint WO method clearly outperforms the other 3 methods and, in the case of 4 products is the Joint WI method which wins the rest of methods.

However, it is important to highlight that these values average many different situations and parameters, including some extreme cases where all the methods behave almost equally. This means that in the operating situations which are more frequent in real applications, the difference may be higher than the one reported by these overall results.

Moreover, in the case of 2 products, from the results of the 100 iterations we create three groups taking into account the probability of buying each product (probability of buying the product 1 is greater, equal or less than probability of buying the product 2) and 3 groups taking into account the stocks for the products (stock for the product 1 is greater, equal or less than stock for the product 2). The results obtained are shown in Figure 2. On one hand, the maximum benefit is obtained for all the methods and results are quite similar when the

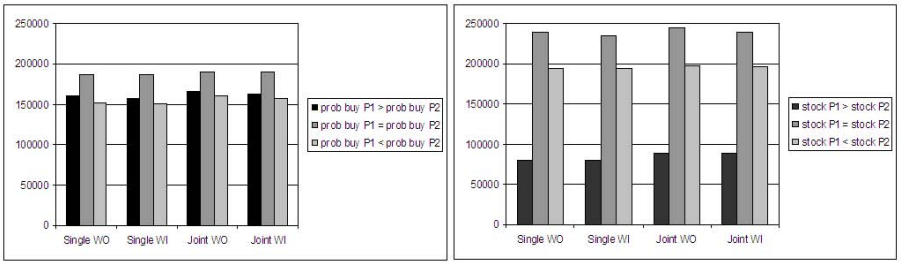


Fig. 2. Left: Variations in probability of buying. Right: Variations in stocks.

popularity (probability of buying) of both products is the same. On the other hand, the maximum benefit is obtained for all the methods and results are quite similar too when both products have the same stock. The results differ between the four methods especially when probabilities or stocks are different.

5 Conclusion

In this paper, we have presented a new framework to address decision making problems where several data mining models have to be applied under several constraints and taking their mutual influence into account. The method is based on the conjunction of simulation with data mining models, and the adjustment of model cutoffs as a result of the simulation with a validation dataset. We have applied this framework to a direct marketing problem, and we have seen that simulation-based methods are better than classical analytical ones.

This specific direct marketing problem is just an example where our framework can be used. Almost any variation of a mailing campaign design problem could be solved (without stocks, with other constraints, non-alternative products, time delays, joint replies, etc.) in some cases with no changes in the presented Petri net and, in the worst case, by just modifying the Petri net that models the constraints and the relations between models. If not only the cutoff is to be determined but also the optimal stock or other important variables, then other techniques, such as evolutionary computation might be used to avoid a combinatorial explosion of the simulation cases. In our example, though, the combinations are not so huge to allow for an exhaustive analysis of all of them.

Out from marketing, we see prospective applicability in many other domains. In particular, the ideas presented here were originated after a real problem we addressed recently in colaboration with a hospital, where resources and data mining models from different services were highly interwoven. Other domains which we are particular familiar with and we plan to use these ideas are the academic world (e.g. university), where we are using data mining models to predict the number of registered students per course each year, but until now we were not able to model the interdependencies between several courses.

References

1. Berry, M., Linoff, G.: Mastering Data Mining: The Art and Science of Customer Relationship Management. John Wiley & Sons, Inc., Chichester (1999)
2. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. JMLR 7, 1–30 (2006)
3. Fagin, R., Kumar, R., Mahdian, M., Sivakumar, D., Vee, E.: Comparing and aggregating rankings with ties. In: PODS 2004. Proceedings of the 32nd symp. on Principles of database systems, pp. 47–58. ACM Press, New York (2004)
4. Ferri, C., Flach, P., Hernández, J.: Improving the AUC of Probabilistic Estimation Trees. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 121–132. Springer, Heidelberg (2003)

5. Forman, G.: Counting positives accurately despite inaccurate classification. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 564–575. Springer, Heidelberg (2005)
6. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE 77(4), 541–580 (1989)
7. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Elsevier, Amsterdam (2005)

Segmentation and Annotation of Audiovisual Recordings Based on Automated Speech Recognition

Stephan Repp[1], Jörg Waitelonis[2], Harald Sack[2], and Christoph Meinel[1]

[1] Hasso-Plattner-Institut für Softwaresystemtechnik GmbH (HPI), P.O. Box 900460, D-14440 Potsdam, Germany
[2] Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2-4, D-07743 Jena, Germany

Abstract. Searching multimedia data in particular audiovisual data is still a challenging task to fulfill. The number of digital video recordings has increased dramatically as recording technology has become more affordable and network infrastructure has become easy enough to provide download and streaming solutions. But, the accessibility and traceability of its content for further use is still rather limited. In our paper we are describing and evaluating a new approach to synchronizing auxiliary text-based material as, e. g. presentation slides with lecture video recordings. Our goal is to show that the tentative transliteration is sufficient for synchronization. Different approaches to synchronize textual material with deficient transliterations of lecture recordings are discussed and evaluated in this paper. Our evaluation data-set is based on different languages and various speakers' recordings.

1 Introduction

Audiovisual recordings in terms of streaming media or video podcasts (vodcasts) are increasingly used for live distance and on-demand lecturing by universities and distance learning institutions. Independent in/from time and place, learners have access to libraries of recorded lectures, often being organized as knowledge bases that offer their content in a well ordered and categorized manner. But, how can appropriate information be retrieved in a large lecture video data base in an efficient way? Manual segmentation of video lectures into smaller units, each segment related to a specific topic, is an accepted approach to find the desired piece of information [7,14,19,17,18].

Traditional multimedia retrieval based on feature extraction cannot be efficiently applied to lecture recordings. Lecture recordings are characterized by a homogeneous scene composition. Thus, image analysis of lecture videos fails even if the producer tries to loosen the scene with creative camera trajectory. A promising approach is based on using the audio layer of a lecture recording in order to get information about the lecture content. Unfortunately, most lecture recordings do not provide optimal sound quality and thus, also the effectiveness of automatic speech recognition (ASR) for the extraction of spoken words suffers even if a speaker-dependent system is used. The raw results of an untrained

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 620–629, 2007.

ASR applied to lecture audio streams are not capable for an accurate indexing. Today, in lectures often text-based presentations such as MS Powerpoint or Portable Document Format (PDF) are used to support the teaching. The best alternative is thus to synchronize slides with presenter speech, which can be extracted using a speech to text engine.

Section 2 of the paper shows current approaches and outlines related work, while section 3 introduces the new segmentation method. Section 4 gives an evaluation of the algorithm and Section 5 concludes the paper with a short summary and an outlook on future work.

2 Related Work

Using speech recognition for indexing the lecture videos is an often used and evaluated procedure [10,7,14]. Text matching is a widely recognized method [12] too. The Text matching method has a direct impact on other fields such as genetic sequence alignment [11]. The main focus of these algorithms is to parse large strings on a block level and to find correct matches to short strings (gene). Much research has been focused on the area of intelligent meeting rooms and virtual classrooms [9,6]. They usually try to minimize error in the speech transcription or recognition stages. Chu and Chen presented an approach for cross-media synchronization [4]. They match audio recordings with text transliterations of the audio recordings based on dynamic programming methods [13]. It differs from our approach. In our case, the content of the presentation slides and the transliteration of audio recording do not match. Chu and Chen make use of explicitly encoded events for synchronization, instead of implicit automated synchronization.

Another non-analytic approach is to synchronize presentation slides by maintaining a log file during the presentation thus keeping track of slide changes. But, most available lecture recordings do neither support desktop recordings nor maintain a dedicated log file. In [19], they synchronize the book sections with word blocks of the transcript. Yamamoto use a sliding window system and a vector model. They measure the precision and the recall of the results. Chen and Heng [2] also propose a method for automatic synchronisation of speech transcripts with presentation slides. After matching the speech transcript with the slides, redundancy and noise are removed by a fitting procedure. Finally, transitions of slides are approximated by using a progressive polynomial fitting.

Our algorithm lies downstream of these related works, assuming speech to text transcription from an out-of-the box commercial speech recognition software. Further we show, how linear text segmentation algorithms [5,3] can be applied to segment lecture video recordings and to map presentation slides to a particular point of time in the video recording. Additionally we suggest a new algorithm, that deals with achieving global synchronization between the transcript and the presentation text. In the synchronization problem, our transcript input is expected to contain a large amount of error from the speech to text process. Our evaluation is divided into two stages:

First, the transliteration generated from the ASR is segmented by standard linear text segmentation algorithms while the text boundaries are assumed as slide transitions. How can standard linear text segmentation algorithm be able to segment the erroneous transcript into coherent segments, without any additional recourse like the slide? Second, the presentation slides are used as an additional resource for the segmenting procedure and for the slide transition detection. We have implemented a vector-space-algorithm based on a sliding window system, as shown in [8,19,1] and a new algorithm for noise removal.

3 Algorithm Description

Synchronization is not a trivial problem because the input data from the speech to text process naturally contains many sources for error and confusion. Errors in this process reside in the speech to text (STT) engines inherent inaccuracy, the inability for us to train accurately the engine, poor sound quality from the lecture videos, and poor enunciation by the lecturer. The synchronization process faces inconsistencies produced by the way that a professor generally lectures and authors his lecture slides. The slides are not a direct script of the lecture, merely an outline. As the lecturer speaks, he elaborates on each topic and discusses related areas, while only referring to key terms from the slides. As a result, the lecture transcript deviates greatly from the slides, and rarely directly matches a continuous sequence of the slide text.

For our algorithm we assume the monotonicity of the progression of the slides. The most real life presentations follow the schedule given in the presentation slides. But, if the presenter skips some slides (slide-gap), our algorithm is able to resynchronize and if the presenter steps back to a previous slide, our algorithm is also able to resynchronize, when the regular schedule continues.

Figure 1 depicts the overall system architecture.

Descriptor Extraction: The generation of keyword descriptors from the presentation slides is realised in the following way. The presentation slides are converted into the portable document format (PDF) and the plain text information is extracted from the PDF. To exclude nonrelevant words (stop words) the

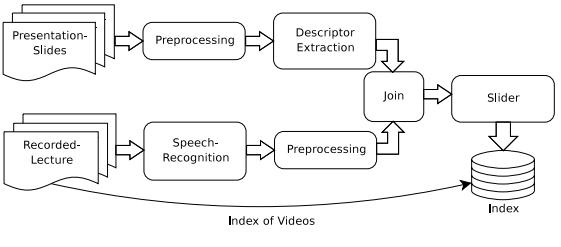


Fig. 1. Overall system architecture

extracted plain text is processed by part-of-speech tagging. Words with high term frequency are removed.

Thus, only words with higher content separatability remain in text. With term frequency and inverse document frequency lesser separating words can be identified and deleted. Generating the keyword stems (word stemming) to enhance the number of hits in the subsequent matching is not suitable, because stemming reduces the content separatability of every word, which is unfavourable for further processing. Hence, for every set of presentation slides we obtain a revised list consisting out of slide numbers and keywords.

Speech Recognition: The lecture video recordings are analysed with an out-of-the-box speech recognition engine. This procedure is already discussed in detail in [16]. The engine needs a short neglectable training phase of 1-3 minutes for adapting the microphone to the ASR. In addition, the extracted slide descriptors complement the engine's vocabulary, but these keywords are not trained by the speakers. The word accuracy is only about 20%-70% per transcript. The transcript consists of a list of words with the corresponding point in time, where the word was spotted in the speaker's flow of words.

Join: The matching procedure works in the following way: (1) Extract all matches between the words from the generated transcript and the slides. This matching results in a tuple (time, slide number, word). (2) Use only the relevant matches for a word that matches at most four times. The parameter four was obtained during a training-phase. It is important to remove datasets with more than four hits per word, otherwise the step function will not be visible. Figure 2 shows the data before and after the filtering. The x-axis represents the time, the y-axis represents the slide numbers. After this procedure a new chronologically sorted list (time, slide number) is generated.

Slider: The purpose of the slider is the removal of the noise and implausible values. The slider is using a bounded-mean-filter. Let N be the number of datasets, s_i the slide number of dataset i with $0 < i < N$ and $0 < s_i \leq S$, with S being the last slide number in the presentation. For the dataset i we define a foregoing environment of k datasets. For these k datasets of the i-th dataset's environment we calculate a mean value $\bar{x}_i$ as:

$$\bar{x_i} = \frac{1}{k} \sum_{j=0}^{k-1} s_{i-j}$$

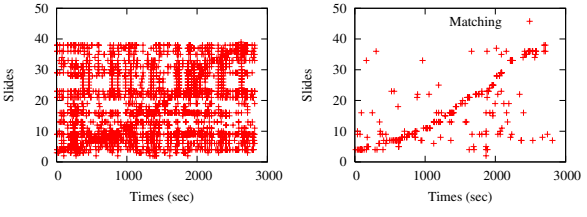


Fig. 2. Matching values before (left) and after filtering (right)

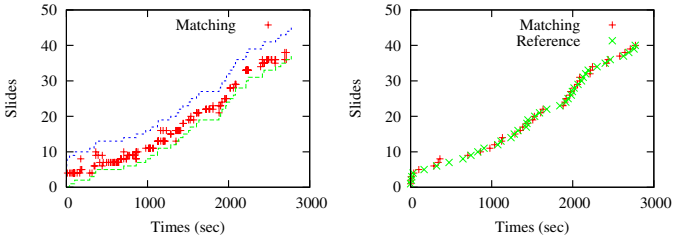


Fig. 3. Values after slider filtering (left), detected slide transitions (right)

The mean value $\bar{x}_i$ is compared to the successive mean, the mean of the i+1-th dataset. If a threshold l satisfies: $\bar{x}_i + l < s_{i+1}$ or $\bar{x}_i - l/2 > s_{i+1}$, the $i+1$-th dataset will be discarded and not used for further calculations. The lower threshold is less than the upper one, thus we get a monotone increasing slope. The algorithm depends on two parameters: the threshold l and the environment k. Parameter tuning was performed on a corpus. The best performance was achieved with k=7 values and a threshold of l=5 slides. The algorithm is implemented recursively: if there exists no match within the given threshold, the algorithm is recursively called with an increase of l.

Figure 3 illustrates the dataset after the use of the noise removal and the both threshold boundaries. Next, the slide steps of the datasets have to be determined. To do this the calculated mean values are used again. Every alteration of the means represents one step. Only one step is allowed per alteration. Thus, the slide transitions can be reproduced. Figure 3 shows the slide starting points and the manually determined reference values.

4 Evaluation

For evaluation we were using the recorded lectures series “Technische Grundlagen des WWW” and “Internet Security - Weaknesses and Targets” of Christoph Meinel from Hasso-Plattner-Institute in Potsdam (Germany) and the recorded lecture series “Semantic Web” of Harald Sack, Friedrich-Schiller-University of Jena (Germany). Our dataset includes two different speakers and two different languages (German and English). Table 1 shows a summary of the dataset’s content.

Table 1. Summary of the lecture series archive

Name	Speaker	Language	Accuracy	Lectures	Words	Duration	Transitions
WWW	Meinel	German	20-70%	24	258190	31h	1069
Semantic	Sack	German	20-55%	10	100437	12h	485
Security	Meinel	English	20-55%	14	130320	15h	448

4.1 Measurement

Mean of Error Rates: The error rates are calculated as difference between the point in time of the reference and the point in time achieved by our algorithm for each slide transition in the course. The meaning of these values is the medial of all differences (i.e. the offset of the time-shift). A positive offset is a positive time shift and a negative offset is a negative time shift. For example $\bar{X} = -100s$ the starting point (or the constant component of the data-set) is 100 seconds after the real start-time of the slide. It is certain that a low offset (the mean $\bar{X}$) stands for higher accuracy of the algorithm. Addinally, we calculate the mean $\bar{Y}$ of all absolute differences. The means of difference are calculated in the following way:

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N}(r_i - c_i) \text{ and } \bar{Y} = \frac{1}{N}\sum_{i=1}^{N} ABS(r_i - c_i)$$

with r_i denoting reference time and c_i calculated time of dataset i.

Standard Deviation: The standard deviation points out how much the data are spread out from the mean $\bar{X}$. If the datasets are close to the mean, the standard deviation is small and the algorithm matches the boundaries very well. Conversely, if many data points are far-off the mean, the standard deviation is large and the algorithm has higher error rates. Higher deviation of values are downgrading the search process. The users are not ready to accept overflowing dissimilarities of the starting time in the search results.

Precision and Recall: For our setting, precision and recall are not adequate for evaluation. The number of slides is well-known and any implemented algorithm would generate a hit for each slide. Thus, the recall would be always 100%. Another argument is the inconvenience of deciding, if a calculated time is a hit or not. This depends on the size of the delay between reference time and calculated time. The acceptable size of delay varies for every video recording and cannot be assigned in an objective way.

WindowDiff: The WindowDiff [15] measurement is used as a standard evaluation measure for text segmentation. WindowDiff does not take into account the time delay between the calculated time and the real boundary time. Even if a promising WindowDiff value is measured, a high discrepancy between the time of the real boundary and the calculated boundary can still exist. Figure 4 is showing texts with ten sentences (boxes) each. Each text box shows a calculated (calc) and a reference (ref) boundary. The calculated boundary in the first example is after the second sentence and in the second example after the fourth

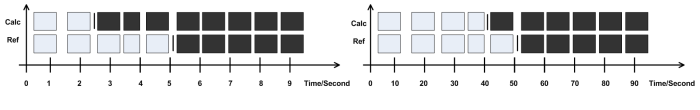


Fig. 4. WindowDiff Example 1. (left) and WindowDiff Example 2.(right)

sentence. The x-axis represents the time, whereas the time in the first example has higher resolution than in the second example. The WindowDiff value of the second example is better than in the first example, but the time delay in the second example is 10 seconds compared to 2.5 seconds in the first example. But, for video retrieval it is preferred to match the exact position in time, than to have an exact text segmentation position. Thus, WindowDiff is not a good choice for video segmentations measurement.

4.2 Comparison of Algorithms

Our algorithm was evaluated in countercurrent with two kinds of algorithms: First, we have compared it with standard linear text segmentation algorithms. The slide transition number is given and a sentence length of 10 words is defined. A slide transition is assumed to be a text boundary and can be detected by the C99 and the LCseg algorithm [3,5]. In this way the number of segments has been provided to the algorithms. A stop-word list and a stemmer both for the German language have been adapted to C99 and LCseg. Additionally, a linear distribution (Linear) of the slide transition during the presentation time has been implemented.

Second, a vector algorithm similar to [19,8] has been implemented. A window slides over the stream of words and each window is matched with the word stems of each slide. The window size has been set to 140 words and the step to 20 words. After standard pre-processing (deletion of stop-words, word stemming, and matrix ranking) the similarity values (cosine measure) between each window

Table 2. Mean, standard deviation, and WindowDiff of experimental results

mean/s $\bar{X}$ ($\bar{Y}$)					
Course	LCseg/s	C99/s	Linear/s	Vector/s	RW07/s
WWW	-31 (674)	6 (215)	85 (204)	28 (311)	27 (52)
Semantic	-63 (742)	184 (321)	179 (293)	205 (663)	5 (89)
Security	-306 (577)	28 (189)	3 (164)	209 (639)	9 (113)
Summary	-100 (669)	54 (235)	89 (217)	111 (470)	18 (74)
Standard Deviation					
WWW	924	294	258	837	**84**
Semantic	1050	359	318	1393	**134**
Security	850	252	217	1211	**163**
Summary	947	311	272	1087	**119**
WindowDiff					
WWW	0.54	0.47	0.53	0.49	**0.46**
Semantic	0.54	**0.50**	0.52	0.52	0.54
Security	**0.49**	0.52	0.51	0.53	0.52
Summary	0.52	**0.49**	0.52	0.51	**0.49**

Table 3. Vector algorithm combined with Slider

	mean/s X (Y)	Standard Deviation/s	WindowDiff
RW07	18 (74)	119	0.49
Vector	111 (470)	1087	0.51
Vector + Slider	-1 (104)	224	0.53

and each slide have been calculated. A detected slide transition maps to the highest similarity between a window and the slide.

4.3 Result

Table 2 shows the mean over the three test corpora and the quantitative results are summarized in final column. The table shows that our RW07 algorithm yields to significantly lower mean and standard deviation than C99, LCseg, Linear and the Vector algorithm for the test corpora. It is not clear, whether the RW07 performs better than the other algorithms in the text segmenting measure WindowDiff. The difference is too low and it is not significant for any statement.

The vector algorithm matches a lot of slides correct. A disadvantage of this algorithm is the large amount of outlier hits (see figure 5). This mismatch results in the loss of the chronological sequence. Only one outlier can produce a large mean and a large standard deviation. The vector algorithm and the increasing slope algorithm (Slider) have been combined to avoid this interference in the result set.

Therefore, the vector algorithm has been modified in the way that the tenth most similar value between a slide and the text windows have been used. The resulting tuple (slide-number, time) of a lecture serves as the input for the Slider.

With this improvement an enhancement in mean and in standard deviation has been achieved.

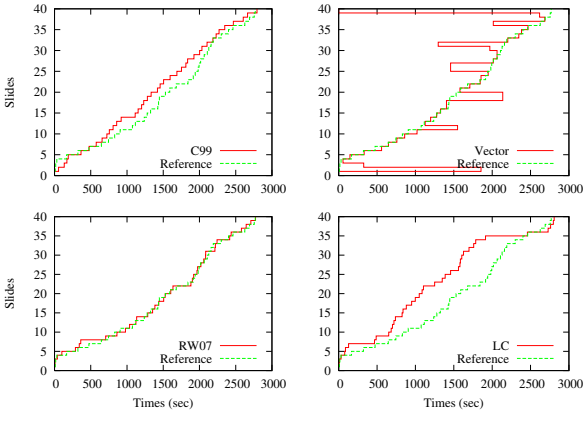


Fig. 5. Overview and error

Compared with RW07 the mean is better, but the more important standard deviation becomes worse. The deviation is approximately 3.5 min, compared to 2 min of the RW07.

5 Conclusion

We have presented a speaker and language independent segmentation and annotation algorithm for synchronization of lecture video recordings and text-based presentations. Keeping in mind that the average duration of a lecture video is about 45 to 90 minutes, a lecturer is speaking about a single slide approximately for only 105 seconds. If it is possible to match the correct slide in the video with an offset of 18 seconds and with a deviation of +/- 119 seconds, then we will find a fuzziness area of +/- 1 slide. For lectures, usually the topics of surrounding slides are very similar and therefore, the fuzziness area has no dramatic effect for the user. It is a significant improvement in accessing the content of video recordings.

The linear text based segmentation approaches (C99, LCseg) are not suitable for the application to erroneous transliteration of lecture video recordings. The vector algorithm is most suitable for a none sequential workflow, where a presentation slide might match to any given point in time of a lecture video recording. But, most lectures follow a sequential order of presentation slides.

The descriptors gained from the presentation slides can be used for MPEG-7 annotations. The application for our algorithm is not only limited to the use in sequentially ordered presentations. All applications with a chronology activity, as e. g. newscasts, theater-plays, or any kind of speech being complemented by textual data could be analysed and annotated with the help of our proposed algorithm. For further refinement, we plan to investigate the effects of adding features such as feature-words to our segmentation and annotation algorithm to achieve even more accurate results.

References

1. Beeferman, D., Berger, A., Lafferty, J.D.: Statistical models for text segmentation. Machine Learning 34(1-3), 177–210 (1999)
2. Chen, Y., Heng, W.J.: Automatic synchronization of speech transcript and slides in presentation. In: ISCAS. Proceedings of the IEEE International Symposium on Circuits and Systems, Circuits and Systems Society, pp. 568–571 (May 2003)
3. Choi, F.Y.Y.: Advances in domain independent linear text segmentation. In: Proceedings of NAACL 2000 (2000)
4. Chu, W.-T., Chen, H.-Y.: Cross-media correlation: a case study of navigated hypermedia documents. In: MULTIMEDIA 2002. Proceedings of the tenth ACM international conference on Multimedia, pp. 57–66. ACM Press, New York, USA (2002)
5. Galley, M., McKeown, K., Fosler-Lussier, E., Jing, H.: Discourse segmentation of multi-party conversation. In: ACL, pp. 562–569 (2003)

6. Gross, R., Bett, M., Yu, H., Zhu, X., Pan, Y., Yang, J., Waibel, A.: Towards a multimodal meeting record. In: IEEE International Conference on Multimedia and Expo (III), pp. 1593–1596 (2000)
7. Haubold, A., Kender, J.R.: Augmented segmentation and visualization for presentation videos. ACM Multimedia, 51–60 (2005)
8. Hearst, M.A.: Texttiling: Segmenting text into multi-paragraph subtopic passages. Computational Linguistics 23(1), 33–64 (1997)
9. Hsueh, P., Moore, J.: Automatic topic segmentation and lablelling in multiparty dialogue. In: First IEEE/ACM workshop on Spoken Language Technology (SLT), Aruba, IEEE Computer Society, Los Alamitos (2006)
10. Hürst, W., Kreuzer, T., Wiesenhütter, M.: A qualitative study towards using large vocabulary automatic speech recognition to index recorded presentations for search and access over the web. In: IADIS Internatinal Conference WWW/Internet (ICWI), pp. 135–143 (2002)
11. Li, M., Ma, B., Wang, L.: Finding similar regions in many sequences. J. Comput. Syst. Sci. 65(1), 73–96 (2002)
12. Myers, G.: A fast bit-vector algorithm for approximate string matching based on dynamic programming. J. ACM 46(3), 395–415 (1999)
13. Ney, H., Ortmanns, S.: Progress in dynamic programming search for lvcsr. Proceedings of the IEEE 88(8), 1224–1240 (2000)
14. Ngo, C.-W., Wang, F., Pong, T.-C.: Structuring lecture videos for distance learning applications. In: ISMSE. Proceedings of the Multimedia Software Engineering, pp. 215–222 (December 2003)
15. Pevzner, L., Hearst, M.A.: A critique and improvement of an evaluation metric for text segmentation. Computational Linguistics 28(1), 19–36 (2002)
16. Repp, S., Meinel, C.: Segmenting of recorded lecture videos - the algorithm voiceseg. In: ICETE. Proceedings of the 1th Signal Processing and Multimedia Applications, pp. 317–322 (August 2006)
17. Repp, S., Meinel, C.: Semantic indexing for recorded educational lecture videos. In: PERCOMW 2006, Washington, DC, USA, pp. 240–245 (2006)
18. Sack, H., Waitelonis, J.: Integrating social tagging and document annotation for content-based search in multimedia data. In: SAAW 2006. Proc. of the 1st Semantic Authoring and Annotation Workshop, Athens (GA), USA (2006)
19. Yamamoto, N., Ogata, J., Ariki, Y.: Topic segmentation and retrieval system for lecture videos based on spontaneous speech recognition. In: EUROSPEECH. Proceedings of the 8th European Conference on Speech Communication and Technology, pp. 961–964 (September 2003)

Mining Disjunctive Sequential Patterns from News Stream

Kazuhiro Shimizu[1], Isamu Shioya[2], and Takao Miura[1]

[1] Dept.of Elect.& Elect. Engr., HOSEI University
3-7-2 KajinoCho, Koganei, Tokyo, 184–8584 Japan
[2] Dept.of Management and Informatics, SANNO University
1573 Kamikasuya, Isehara, Kanagawa 259–1197 Japan

Abstract. Frequent disjunctive pattern is known to be a sophisticated method of text mining in a single document that satisfies *anti-monotonicity*, by which we can discuss efficient algorithm based on APRIORI. In this work, we propose a new *online and single-pass algorithm* by which we can extract current frequent *disjunctive patterns* by a weighting method for past events from a *news stream*. And we discuss some experimental results.

1 Motivation

Recently much attention has been focused on quantitative analysis of several features of text thus a new approach, called *text mining*, has been proposed to extend traditional methodologies[13]. Text mining approach comes from *frequency* and *co-occurrence* among text[1], the former means important words arise many times while the latter says related words occur at the same time[10].

On the other hand, recently much attention has been focused on data mining on *data stream* which is huge amount dynamic data on network[6]. Generally we see that data stream has some characteristics;(1)huge amount, (2)high speed, (3)changing data distribution, (4)continuous. Therefor we like algorithms which should be able to output approximate solutions for any time and any stream during the whole process and rapid analysis with limited calculation resources. That's why we cannot apply traditional methods for this problem[6].

Especially data stream such as news articles and broadcast transcription is called *news stream*. Generally we know that issue interval of news articles are irregular. So we assume that time stamp of news articles correspond to content-time of news articles. Under the news stream environment, we see that news articles always occur, and collections of news articles grow incrementally with new ones. We usually have interests in recent ones, i.e., new ones are more important than old.

There have been important research results obtained based on *anti-monotonicity* such as APRIORI[2,4]. Generally we can avoid vast range of search but not enough to sequence data, since in text mining the property doesn't hold

[1] Text means *sequence data*, i.e., an ordered list of information.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 630–642, 2007.

any more and we can't apply APRIORI technique any more. An interesting approach of *disjunctive patterns* has been proposed with a new counting method for frequency to get efficient pattern mining algorithm on single sequence data. The counting method satisfies anti-monotonicity thus we can obtain efficient mining algorithm based on APRIORI[11].

Generally end users got used to try the algorithms many times. This is because they want to get and compare several results with various parameter values through trial and error. Therefor they see that the algorithms take very long time for final results with many exhaustive scan. Moreover they need to re-calculate to get another results for independent processes if they change several parameters. To overcome this issue, *Online Analysis* has been proposed[1]. Particularly under data stream environment, online analysis approach is very important for all but impossible re-calculation. In the previous work, we proposed a new method of extracting disjunctive patterns under static data environment by using online analysis approach[12].

Let us see some related works. As data mining algorithm to data stream, we know *Lossy Counting*[9]. This is an online and single pass algorithm which generates guaranteed frequent itemsets quickly for transaction data stream. Moreover this has many improvement[8] used for large sensor networks. However it doesn't consider adaptability to time progress. There have been several weighting methods proposed so far under data stream environment, such as *decay function*[5,14] and *Tilted-Time Window*[3]. However, all these don't assume data mining of news stream.

In this investigation, we define a new online and single-pass algorithm by which we can extract current frequent disjunctive patterns by a weighting method for past events from a news stream. We introduce disjunctive patterns and a counting measure in section 2. In section 3, we define a problem of mining disjunctive patterns from news stream. Section 4 contains how to construct condensed information, called *Disjunctive Pattern Lattice* (DPL) and how to extract frequent disjunctive patterns from DPL. Section 5 contains some experimental results. We conclude our work in section 6.

2 Disjunctive Patterns

In this work, we consider a word as a unit, called an *item*. Any element in an itemset $I = \{i_1, .., i_L\}, L > 0$ is called *alphabet*, and a *sequence data* (or, article) $S = s_1...s_m$ $(\infty > m > 0)$ is an ordered list of items, m is called a *length* of S, and S is called m-sequence. Note an item may appear many times in S.

A *disjunctive pattern* (or just a pattern) p is a form of $t_1t_2...t_n$ where each t_i is an alphabet a or a disjunction $[a_1a_2...a_m], m > 0$, each a_j is a distinct alphabet. Given two patterns $p = t_1t_2...t_n$ and $q = v_1v_2...v_m, m \leq n$, we say q is a *sub-pattern* of p, denoted by $q \sqsubseteq p$, if there exist $1 \leq j_1 < ... < j_m \leq n$ such that each v_k corresponds to t_{j_k} (denoted by $v_k \sqsubseteq t_{j_k}$ if no confusion arises) satisfying:

If v_k is an alphabet a, $t_{j_k} = a$ or t_{j_k} is a disjunction containing a
If v_k is a disjunction $[a_1 a_2 ... a_m]$, we have both $t_{j_k} = [b_1 b_2 ... b_l]$ and $\{a_1, .., a_m\} \subseteq \{b_1, .., b_l\}$

Example 1. ac is a sub-pattern of $abcd$. Similarly $[ac]$ is a sub-pattern of $[abcd]$, bd is a sub-pattern of $[ab][cd]de$, b is a sub-pattern of $[ab]$, and "ac" is a sub-pattern of $[ab][cd]$. However, ab is not a sub-pattern of $[ab]$, nor $[ab]$ is a sub-pattern of ab.

We say a pattern $p = t_1 t_2 ... t_n$ *matches* a sequence $S = c_1 c_2 ... c_m$ if t_1 is an alphabet a_1, there exist $t_1 = a_1 = c_{i_1}, 1 \leq i_1 \leq m$ and the sub-pattern $t_2 ... t_n$ matches $c_{i_1+1} ... c_m$, and if t_1 is a disjunction $[a_1 a_2 ... a_m]$, there exists a permutation $a_{j_1} ... a_{j_m}$ of $a_1, ..., a_m$ that matches $c_1 ... c_{i_1}$, and the subpattern $t_2 ... t_n$ matches $c_{i_1+1} ... c_m$.

Example 2. Assume S is $\mathtt{a_1 a_2 b_3 b_4 b_5 a_6}$ where each suffix shows its position in the occurrence. A pattern a matches S 3 times($\mathtt{a_1}, \mathtt{a_2}, \mathtt{a_6}$), ab 6 times ($\mathtt{a_1 b_3}, \mathtt{a_1 b_4}$, $\mathtt{a_1 b_5}, \mathtt{a_2 b_3}, \mathtt{a_2 b_4}, \mathtt{a_2 b_5}$) and $[ab]$ 9 times. Note we can see more frequency by $[ab]$.

Given a sequence data S, a function $\mathcal{M}_S$ from patterns to non-negative integers satisfies *Anti Monotonicity* if for any patterns p, q such that $q \sqsubseteq p$, we have $\mathcal{M}_S(q) \geq \mathcal{M}_S(p)$. In the following, we assume some S and we say $\mathcal{M}$ for $\mathcal{M}_S$. Given $\mathcal{M}$ and an integer $\sigma > 0$ (called *minimum support*), A pattern p is called *frequent* if $\mathcal{M}(p) \geq \sigma$. If $\mathcal{M}$ satisfies anti-monotonicity, for any q such that $\mathcal{M}(q) < \sigma$, there is no frequent p such that $q \sqsubseteq p$. By using this property, we can reduce search space to extract frequent patterns. However, it is not easy to obtain $\mathcal{M}$ satisfying anti-monotonicity. For example, "the number of matching" is not suitable as $\mathcal{M}$ as shown in EXAMPLE 2. There have already been proposed a counting method that satisfies anti-monotonicity[11].

The first idea is called *head frequency*. Given a sequence $S = s_1 s_2 ... s_r$ and a pattern p of $t_1 t_2 ... t_n$, we define a *head frequency* $H(S, p)$ as $H(S, p) = \sum_{i=1}^{r} Val(S, i, p)$ where $Val(S, i, p)$ is 1 if the following holds, and 0 otherwise:

Let $S(i)$ be a suffix of S from i-th position, i.e., $S(i) = s_i ... s_r$. If t_1 is an alphabet a, we have $s_i = a$ and $t_2 t_3 ... t_n$ matches $S(i+1)$. And if t_1 is a disjunction $[a_1 a_2 ... a_m]$, there exists j such that $s_i = a_j$ (for instance, $j = 1$), and $[a_2 a_3 ... a_m] t_2 ... t_n$ matches $S(i+1)$.

Intuitively $H(S, p)$ describes the number of matching of p from the heading of S or its suffix. However, it is known that the head frequency $H(S, p)$ doesn't satisfy anti-monotonicity. Note that this counting ignores matching appeared in the subsequent sequence. That's why we introduce a new counting $D(S, p)$, called *total frequency*, which means the minimum $H(S, q)$ for any $q \sqsubseteq p$: $D(S, p) = \mathrm{MIN}\{H(S, q) | q \sqsubseteq p\}$

In fact, we have $D(S, p) = \mathrm{MIN}\{H(S, p), D(S, p(2))\}$ where $p(2)$ means a sub-pattern of p that has the length pf p minus 1. And also it is enough to calculate the ones for all the suffixes of p in ascending order of length. Then we can show that $D(S, p)$ satisfies anti-monotonicity. Note there are only n suffixes of p thus we can reduces search space dramatically[11].

Example 3. Let S be caabbbc. If $p = ab$, we have $H(S,p) = 2$ and $D(S,p) = 2$. If $p = ac$, we have $H(S,p) = 2$ and $D(S,p) = 2$. And if $p = [ac]$, we see $H(S,p) = 3$ and $D(S,p) = 2$ while p matches S 4 times. In these cases, sub-patterns (i.e., a, c) of ac and $[ac]$ appear more interspersed in S and this is why *total frequency* is different from *head frequency* and the number of matching.

In this work we assume *single* document and the frequency of words corresponds to the importance directly. However we examine a test collection containing multiple articles in this experiment which means that the longer articles become the more words appear, and the frequency doesn't always mean the importance. For this reason we give some *weights* to sequences to avoid the effect of the length. In this work, we introduce *Term Frequency* (TF) for frequent words[7]. For a pattern p and a pattern length of p n, denoted by $H_w(S,p)$, is defined as $H_w(S,p) = \frac{H(S,p)}{|S|-(n-1)}$ where $|S|$ means the number of words in S. Then we give *weightened total frequency*, $D_w(S,p)$, defined as $D_w(S,p) = \mathrm{MIN}\{H_w(S,p), D_w(S,p(2))\}$ In our new method which is proposed at following sections, we utilize $H_w(S,p)$ and $D_w(S,p)$ as both head frequency and total frequency.

3 Problem Definition

Let us define what mining disjunctive patterns from news stream means. Given *3 inputs* of news stream NS, minimum support σ and error bound ϵ, we want to find all frequent patterns which have frequency $(\sigma - \epsilon)$ or above when it is current time $time_{now}$D

Putting it into more specific description, we define an infinite ordered list of sequence data $S_1 ... S_i ... S_j ...$, called news stream NS, and S_i means i-th sequence data in NS. Each S_i in NS has time stamp $time_i$, and if $i \leq j$ then $time_i \leq time_j$. In this work, we consider sequence data $S_{i_1} ... S_{i_{wsize}}$ with a common $time_i$ as a single sequence data S_i. Window size $wsize$ means the number of sequence data which can be calculated at once. Current time $time_{now}$ is a query time which has different interpretations at each point. In this section, a text pattern is called *frequent pattern* if the frequency is more than a threshold σ.

Most of algorithms which take data stream have to consider huge amount data and long term analysis. Generally, they output approximate frequent itemsets to avoid a lot of computation cost. In this investigation, we propose a new algorithm which is based on *Lossy Counting*[9] for mining disjunctive patterns from news stream. Lossy Counting constructs a data structure by using each item's estimate frequencies and maximum errors. The constructed data structure satisfies two important properties: *Property1* : If an item is in a data structure, then its exact frequency is more than its estimate frequency and less than the sum total of its estimate frequency and maximum error. *Property2* : If an item isn't in a data structure, then its exact frequency is less than an error bound.

The solutions from the constructed data structure at chosen random time are guaranteed frequent itemsets by an error bound ϵ, in other words, they have frequencies which are more than $\sigma - \epsilon$.

Here is a new online and single-pass algorithm which consists of 3 steps.

(1) Generating a collection of candidate patterns from sequence data.
(2) Constructing a disjunctive pattern lattice from a generated collection of candidate patterns.
(3) Extracting a collection of frequent patterns from a constructed disjunctive pattern lattice.

The algorithm goes through step (1) and (2) in a repeated manner whenever new article comes in from news stream. Or the algorithm generates a collection of frequent patterns by step (3) if users want. A collection of candidate patterns are generated efficiently using *list of positions* at step (1), and a particular data structure called *disjunctive pattern lattice* are also constructed at step (2). Then we extract frequent patterns quickly from a constructed disjunctive pattern lattice at step (3).

Here we assume that frequencies on the latest sequence are the most important and we give a *weight* based on timestamp of sequence data, denoted by ω. In the algorithm, we give a *decay function* ω as a weight when we access each node in a disjunctive pattern lattice at step (2) and (3), and consequently, frequencies on sequence data which have old timestamps are decreased exponentially: $\omega = u\lambda^{(time_{now} - time_{last})}$ where $time_{last}$ means last update time of each pattern in a disjunctive pattern lattice and λ means the decay constant $0 \leq \lambda \leq 1$. If λ becomes smaller, then ω becomes smaller too and frequencies are decreased more. Also u describes the unit step function such that $u = 1$ if $(time_{now} - time_{last} \leq \tau)$, and $u = 0$ otherwise, where τ means valid time.

4 Algorithm

Let us discuss how to generate a collection of *candidate patterns* from sequence data S in NS. The main idea is that we keep a list of pairs of *head* and *tail* positions to each frequent pattern p where "head" means starting position of p in S and "tail" means ending position. Let $P_p(head, tail)$ be the two positions of p in S. Then we define a *list of positions of* p (in S), $list_p$, as $\{P_{p_i}(head, tail), ..., P_{p_{max}}(head, tail)\}$ where max means the head frequency and all the head values are distinct by definition. Here is the algorithm to generate a collection of n -candidate patterns C_n from S.

Input: a sequence data S
Output: a collection of n -candidate patterns C_n
Algorithm:
if $n = 1$**:** After scanning S, obtain all 1-frequent patterns and generate the corresponding nodes and the arcs to C_n. Let n be 2.
if $n > 1$**:** By examining the lists, obtain all n-frequent patterns and add the nodes and the arcs to C_n. Let n be $n + 1$.

In this algorithm, we obtain $list_{pq}$ of an n-frequent pattern pq by examining $list_p$ and $list_q$ of $(n-1)$-frequent patterns p, q, described below:

Input: $list_p$, $list_q$
Output: $list_{pq}$
Algorithm:

1. By examining $list_p$ and $list_q$, we find all the pairs of $P_{p_i}(head)$ and $P_{q_j}(tail)$ such that $P_{p_i}(tail) < P_{q_j}(head)$, and add the pair to $list_{pq}$ as $(head, tail)$ of k-th element P_{pq_k}.
2. After completing the match, output $list_{pq}$.

Eventually we have $list_p, list_q$ and $list_{pq}$:

$$list_p = \{P_{p_i}(head, tail), ..., P_{p_{imax}}(head, tail)\}$$
$$list_q = \{P_{q_j}(head, tail), ..., P_{q_{jmax}}(head, tail)\}$$
$$list_{pq} = \{P_{pq_k}(P_{p_i}(head), P_{q_j}(tail))|P_{p_i}(tail) < P_{q_j}(head)\}$$

Example 4. Let S be `caabbbc`, and a, b, c be 3 patterns (of 1 item). We get 3 lists $list_a = \{P_{a_1}(2,2), P_{a_2}(3,3)\}$ $list_b = \{P_{b_1}(4,4), P_{b_2}(5,5), P_{b_3}(6,6)\}$, and $list_c = \{P_{c_1}(1,1), P_{c_2}(7,7)\}$. Also we get 2 lists $list_{ab} = \{P_{ab_1}(2,4), P_{ab_2}(3,4)\}$ and $list_{[ac]} = \{P_{[ac]_1}(1,2), P_{[ac]_2}(2,7), P_{[ac]_3}(3,7)\}$.

Given a set I of items and a single sequence S, to examine online analysis for disjunctive patterns, we make up a lattice over a power set 2^I, called *Disjunctive Pattern Lattice* (DPL), as shown in a figure 1. Formally a DPL is a rooted, acyclic directed graph (V, E) where V means a finite set of nodes with labels and E means a set of arcs such that $E \subseteq V \times V$. In a DPL, there exists only one node, called a root, without any label. A node $v \in V$ with the distance

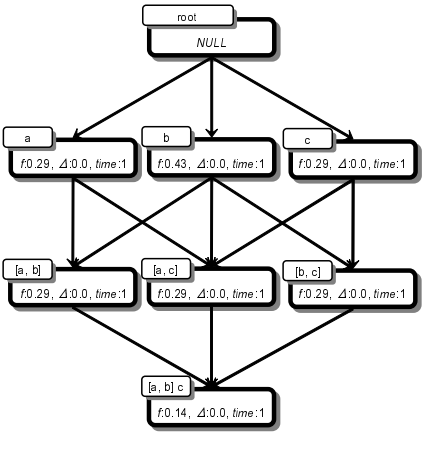


Fig. 1. DPL of `cabcbba` with $\epsilon = 0.1$

$n \geq 0$ from the root corresponds to a candidate pattern of n items (called an n-candidate pattern) with the *label* $(f, \Delta, time_{last})$ where f means its estimated pattern frequency which has been calculated until $time_{last}$, Δ means its pattern's maximum error and $time_{last}$ means last update time of its node. There is an *arc* $e\langle v_1, v_2\rangle \in E$ if and only if n-candidate pattern v_1 has v_2 as an $(n+1)$-candidate sub-patterns for some n.

Example 5. Let $S \in NS$ be a sequence `cabcbba`, $time = 1$ and $\epsilon = 0.1$. Then we have the DPL as in a figure 1.

Here is the algorithm to append $p \in C_n$ to DPL $\mathcal{D}$ and update/delete a node of p in $\mathcal{D}$ from sequence data S_i.

Input: a collection of candidate patterns C, error bound ϵ
Output: DPL $\mathcal{D}$
Operations:

***Insert*:** Insert a node v_p with a label $(\sum D_w(S_i, p), \epsilon N_{last}, time_i)$ to $\mathcal{D}$ if $\exists v_p \notin \mathcal{D}$ and $\sum D_w(S_i, p) \geq \epsilon(N_{all} - N_{last})$.

***Update*:** Update a label of a node v_p in $\mathcal{D}$ to $(\sum D_w(S_i, p) + \omega f, \Delta, time_i)$ if $\exists v_p \in \mathcal{D}$ and $(\sum D_w(S_i, p) + \omega f + \Delta \geq \epsilon N_{all})$.

***Delete*:** Delete v_p and all of v_p's super nodes in $\mathcal{D}$ if $\exists v_p \in \mathcal{D}$ and $(\sum D_w(S_i, p) + \omega f + \Delta < \epsilon N_{all})$.

Note N_{all} means all of processed sequence data which include S_i and N_{last} means processed sequence data $N_{all} - wsize$ without S_i. Also let us note $\sum D_w(S_i, p)$ describes the summation of total frequencies $D_w(S, p)$ from $N_{last}+1$ to N_{all}, that is, $\sum D_w(S_i, p) = \sum_{l=N_{last}+1}^{N_{all}} D_w(S_{i_l}, p)$ Also $time_{now}$ is used to obtain ω in this step which corresponds to a timestamp $time_i$ of S_i.

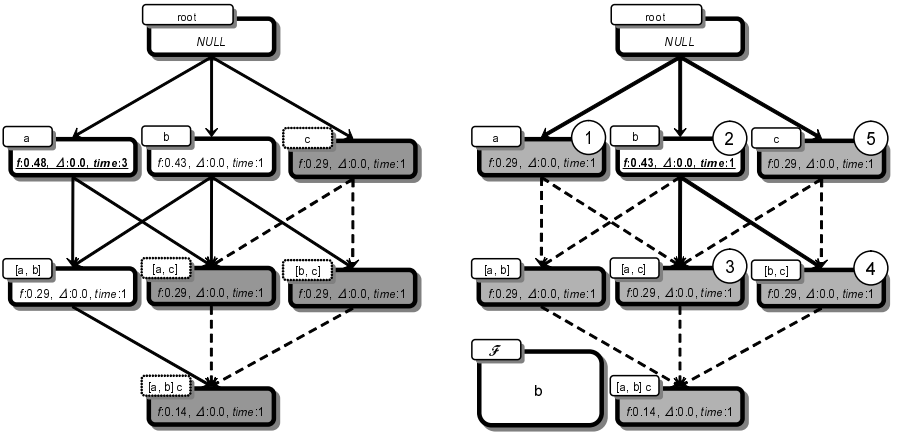


Fig. 2. Updating/Deleting, and Extracting Frequent Patterns from a DPL ($\sigma = 0.5$)

Example 6. Now we update a node v_a in the DPL of a figure 1. We are given $\epsilon = 0.1$, $N_{all} = 3$, $time_{now} = 3$, $wsize = 1$, $\lambda = 0.98$, $\tau = 3$ and $\sum D_w(S, a) = 0.2$. Then $\omega = 0.96$ and a weighted frequency of p is $\sum D_w(S, p) + \omega f + \Delta = 0.48$. Also $\epsilon N_{all} = 0.3$ and we see that v_a's label is updated $(f : 0.48, \Delta : 0, time_{last} : 3)$. On the other hand, we assume that v_c is deleted from the DPL. In this case, we can see that it's supper patterns $v_{[a,c]}, v_{[b,c]}, v_{[a,b]c}$ are deleted as well. After operating nodes, we get the DPL as in a figure 2(left).

Now let us describe how to extract frequent patterns from DPL. Basically we apply *depth first search* algorithm to DPL. Note $time_{now}$ is used to obtain ω in this step.

Input: DPL $\mathcal{D}$, minimum support σ, error bound ϵ
Output: a collection of frequent patterns $\mathcal{F}$
Algorithm:

1. Start with the root in DPL, find 1-frequent patterns which satisfies $(\sigma - \epsilon)N_{all} \leq \omega f$, push them to a stack K and add them to $\mathcal{F}$.
2. For each v popped from K, go to v' through an arc $e\langle v, v' \rangle$ in DPL. Push v' to K if we have never visited v' and add v' to $\mathcal{F}$ if v' satisfies $(\sigma - \epsilon)N_{all} \leq \omega f$. Goto (2).
3. Discard v' if we have already visited v', and goto (2)
4. After examining all the K elements, we output $\mathcal{F}$.

Example 7. We extract all the frequent patterns which have frequencies $(\sigma - \epsilon)N_{all} = 0.4$ from a DPL in a figure 1. Now, we are given $\sigma = 0.5$, $\epsilon = 0.1$, $N_{all} = 1$, $time_{now} = 2$, $\lambda = 0.98$ and $\tau = 3$. According to the traversal as in a figure 2 (right), we obtain $\mathcal{F}$.

5 Experimental Results

5.1 Preliminaries

In this section we examine 4 kinds of experiments containing scalability, efficiency of construction of DPL and extracting frequent patterns from DPL, and space utilization for candidate patterns in DPL and the flexibility of time progress. We also analyze about the validity of our experimental results and deliberate about the weighting method in our method. In this experiment, we discuss only 1 level of disjunctive patterns where length are 5 or less such as `a, [ab], [ab]c, [abc]d, [abcd]e` . Also we assume disjunctive sub-pattern appears at most one time in any patterns.

Here we examine *Reuters-21578 text categorization test corpus* as a test collection. The collection consists of 21,578 news articles in 1987 provided by Reuters kept in time order. We have selected 2200 articles as 22 datasets which include 100 articles which have a common time stamp per dataset in time order and cleaned them by stemming and removal of stop-words [7]. We take "day" which is relative time for the first dataset as a time unit of time stamps through our experiments. Here are some examples of the results. `shower continu week bahia`

Table 1. Details of experimental data

dataset#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
time[day]	0	5	7	11	14	15	19	21	25	26	28	32	34	36	40	41	46	95	96	113	235	236
diff.[day]	-	5	2	4	3	1	4	2	4	1	2	4	2	2	4	1	5	49	1	17	122	1

`cocoa zone allevi drought earli januari improv prospect temporao humid level restor...` Details of our experimental data are found in a table 1.

In our experiments, we take 4 kinds of the decay constants λ which are 1.00(without weighting), 1.00_7(valid for 1 week), 0.98(approx. $\omega = 0.5$ of 1 month past) and 0.91(approx. $\omega = 0.5$ of 1 week past).

5.2 Results

In the first experiment, we give 0.005, 0.008 and 0.010 as error bound ϵ, 0.98 as a decay constant λ, and examine the construction time (in seconds) of a DPL every 50 articles in the experimental data. We show the scalability of our method in a figure 3(left).

In the experiment 2, we examine the frequency of the operations which are "Insert", "Update" and "Delete" and the execution time for the construction of DPL for each threshold, (1)error bound ϵ, (2)the number of simultaneous processed articles and (3)decay constant. In the case of (1), we give 0.005, 0.008 and 0.010 as ϵ with the number of simultaneous processed articles. In the case of (2), we give 25, 50 and 100 as the number of simultaneous processing articles with 0.008 as ϵ. In the case of (3), we give 0.008 as ϵ with 50 as the number of simultaneous processed articles. Both tables in a table 2 and a table 3(upper) illustrate the experimental results on each case.

A table 3 (bottom) contains the results of the experiment 3. We give $\epsilon = 0.005$, $\lambda = 0.98$, 100 as the number of simultaneous processed articles and give 0.005, 0.006, ...,0.010, 0.015, 0.020, 0.030 as minimum support σ and examine

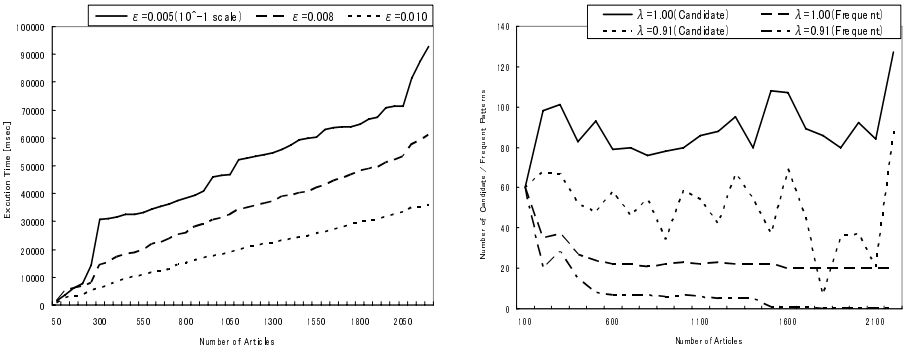


Fig. 3. Scalability; the number of nodes in DPL and frequent paterns

Table 2. Constructing a DPL for ϵ (upper) and λ (bottom)

λ	1.00			1.00_7			0.98			0.91		
ϵ	.005	.008	.01	.005	.008	.01	.005	.008	.01	.005	.008	.01
Insert	1301	461	258	1360	483	275	1388	487	281	1453	521	296
Update	1579	614	364	1368	530	316	1230	486	289	903	359	223
Delete	1125	400	223	1206	425	241	1233	429	247	1293	462	261
Time[sec]	1041.1	67.8	36.2	1054.3	66.8	36.9	929.1	61.2	36.1	738.6	56.5	35.7

λ	1.00	1.00_7	0.98	0.91	λ	1.00	1.00_7	0.98	0.91
Insert	461	483	487	521	Delete	400	425	429	462
Update	614	530	486	359	Time[sec]	67.8	66.8	61.2	56.5

Table 3. Constructing a DPL on number of articles and Execution Time (sec.)

λ	1.00			1.00_7			0.98			0.91		
# of articles	25	50	100	25	50	100	25	50	100	25	50	100
Insert	2597	461	172	2618	483	187	2626	487	188	2643	521	203
Update	1450	614	266	1328	530	221	1210	486	196	1024	359	124
Delete	2494	400	134	2524	425	156	2532	429	158	2549	462	172
Time[sec]	224.7	67.8	48.3	239.1	66.8	50.7	208.9	61.2	44.3	196.4	56.5	39.4

σ	.005	.006	.007	.008	.009	.01	.015	.02	.03
Time[msec]	47	46	32	31	31	31	31	31	16
# of patterns	101	101	72	51	39	37	7	5	2

Table 4. The number of frequent patterns for past days

Past days	0	7	14	21	28	35	42	49	56	63	70	Past days	0	7	14	21	28	35	42	49	56	63	70
$\lambda = 1.00$	37	37	37	37	37	37	37	37	37	37	37	$\lambda = 0.98$	37	31	20	18	8	7	7	6	5	4	2
$\lambda = 1.00_7$	35	0	0	0	0	0	0	0	0	0	0	$\lambda = 0.91$	28	7	2	0	0	0	0	0	0	0	0

the execution time (in seconds) to extract all the frequent patterns from 300 articles in the experimental data.

In the experiment 4, we give $\epsilon = 0.005$, $\sigma = 0.010$ and 100 as the number of simultaneous processed articles and we examine the comparison of the number of frequent patterns and candidate patterns which mean nodes in a DPL with the cases of $\lambda = 1.00$ and $\lambda = 0.91$. Also we examine the number of frequent patterns which are extracted from constructed DPL by 300 articles every 7 days under same conditions. A figure 3(right) and a table 4 illustrate the results.

5.3 Discussion

Let us discuss what the results mean. In the experiment 1, the reader sees the scalability of our method. In our method, 2 times bigger ϵ causes 25.7 times longer execution time. The result as a whole becomes approximately linear

increase for any ϵ. However, in a first part(0 to 300 articles) of the result, the efficiency is much worse by perpetual inserting nodes to a DPL. Also in a final part(2050 to 2200 articles) of the result, the efficiency is much worse as well. This is because our method gets the effect of time progress which becomes bigger with smaller ϵ.

Through experiment 2, the reader can see the efficiency of construction of DPL. In the case of (1), we see that 2 times bigger ϵ causes 28.8 times longer execution time and 4.74 times larger the number of all operations which are "Insert", "Update" and "Delete" when we give 1.00 as λ. Similarly we need 20.7 times longer execution time and 4.68 times bigger costs when we give 0.91 as λ. This is because our method generates more candidate patterns with smaller ϵ. In the case of (2), we see that 4 times bigger the number of simultaneous processed articles causes 4.65 times shorter execution time and 11.3 times smaller the number of all operations when we give 1.00 as λ. Similarly we need 4.99 times shorter execution time and 12.5 times smaller costs when we give 0.91 as λ. Because our method can execute the operations which is like a batch processing for DPL by simultaneous processing of articles. In the case of (3), we see that smaller λ reduces the costs of construction, more specifically, the number of "Insert" and "Delete" is increased and "Update" is decreased. This is because our method updates the candidate patterns in DPL frequently for smaller λ.

Experiment 3 shows how efficiently our method works. Our method takes more bigger execution time when we give smaller minimum support σ. However we think that it isn't much worse. For example, 6 times bigger σ causes 2.9 times longer execution time. This is because our method scans the collection only once to extract frequent patterns.

In the experiment 4, we examine space utilization for candidate patterns in DPL and the flexibility of time progress.

As for space utilization for candidate patterns, our method gets worse efficiency for remaining candidate nodes in DPL with smaller λ, an average of 28% with 1.00 and 17% with 0.91. This is because, in our method, candidate patterns in DPL become infrequent quickly for bigger weight with smaller λ.

As for the flexibility of time progress, after the 1600 articles, we see that the number of candidate patterns and frequent patterns in DPL are reduced markedly when we give 0.91 as λ. Moreover, in a table 4, we can see that our method takes account of newer patterns in DPL with smaller λ, the number of extracted frequent patterns from DPL are 84% after 1 weak and 54% after 3 weeks with 0.98, 25% and 0% with 0.91 and both 0% with 1.00_7. From this results, we think that our method deletes old candidate patterns from DPL with given weight.

Finally, we analyze about the validity of our experimental results and deliberate about the weighting method in our method. In our experiments, we can see that smaller λ improves the execution efficiency of our method. In other words, we think that our method pays a consideration of that. Therefor we construct a

Table 5. The coverage of dominant patterns with 2200/1700 articles

Cover[%]	said	mln	dlr	pct	year	billion	campani
$\lambda : 1.00$	100/100	100/100	100/100	100/100	100/100	100/100	100/100
$\lambda : 1.00_7$	77.3/100	77.3/100	77.3/100	77.3/100	77.3/100	100/100	77.3/100
$\lambda : 0.98$	90.9/100	86.4/100	77.3/100	77.3/100	72.7/94.1	40.0/40.0	77.3/100
$\lambda : 0.91$	77.3/100	63.6/82.4	63.6/82.4	45.5/58.8	18.2/23.5	6.7/6.7	18.2/23.5
Cover[%]	bank	share	ct	net	loss	sale	*Total*
$\lambda : 1.00$	100/100	100/100	100/100	100/100	100/100	100/100	*100/100*
$\lambda : 1.00_7$	61.5/100	77.3/100	77.3/100	77.3/100	100/100	100/100	*78.6/100*
$\lambda : 0.98$	0/0	72.7/94.1	77.3/100	50.0/64.7	0/0	33.3/33.3	*67.1/82.6*
$\lambda : 0.91$	0/0	18.2/23.5	45.5/58.8	13.6/17.6	0/0	33.3/33.3	*33.3/44.6*

DPL with 0.005 as ϵ and 100 as the number of simultaneous processed articles. After that we extract frequent patterns from a constructed DPL with 0.01 as σ, select dominant patterns which have frequency of 81% for all of extracted frequent patterns' frequency, and compare the coverage of dominant patterns for any λ. We give the coverage (percentage) of dominant patterns *Cover* as $\frac{\#\ of\ extraction\ of\ patterns\ for\ any\ \lambda}{\#\ of\ extraction\ of\ patterns\ for\ \lambda = 1.00} \times 100$

A table 5(left) shows the coverage of dominant patterns for whole experimental data. As shown, smaller λ causes reduction of the coverage, 33.3% with 0.91. That is, we obtain efficiency of our method by reduction of the coverage. Therefore, we limit local appearance of frequent patterns by using the weighting method. That's reason why we can improve the execution efficiency of our method.

From a table 1, we can see that there are a lot of time interval from 1800 to 2200 of the experimental data, that is to say, the update interval of the patterns in DPL becomes longer. A table 5 (right) shows the coverage of dominant patterns for 1700 articles of the experimental data. By this table and a table 5 (left), we get 21.4% smaller coverage with 1.00_7 as λ. That is, we can extract frequent patterns of the whole data in relevant execution time by using the weighting method.

6 Conclusion

In this investigation we have proposed an online and single-pass algorithm by which we can extract current frequent disjunctive patterns by a weighting method for past events from a news stream. We have introduced a sophisticated structure, DPL, discussed how to construct DPL and given a decay function as a weighting method for time progress. Also, by experimental results, we have shown that our method is effective for extraction of disjunctive patterns and construction in real data.

References

1. Aggarwal, C.C., Yu, P.S.: Online Generation of Association Rules. ICDE, 402–411 (1998)
2. Agrawal, R., Srikant, R.: Fast Algorithm for Mining Association Rules. In: proc. VLDB, pp. 487–499 (1994)
3. Chen, Y., Dong, G., Han, J., Wah, B.W., Wang, J.: Multi-dimensional regression analysis of time-series data streams. In: International Conference on Very Large Databases, pp. 323–334 (2002)
4. Goethals, B.: A Survey on Frequent Pattern Mining, Univ.of Helsinki (2003)
5. Ishikawa, Y., Kitagawa, H.: An Improved Approach to the Clustering Method Based on Forgetting Factors. DBSJ Letters 2(3), 53–56 (2003) (in Japanese)
6. Jian, N., Gruenwald, L.: Research Issues in Data Stream Association Rule Mining, SIGMOD Record 35-1, pp. 14–19 (2006)
7. Grossman, D., Frieder, O.: Information Retrieval – Algorithms and Heuristics. Kluwer Academic Press, Dordrecht (1998)
8. Loo, K.K., Tong, I., Kao, B.: Online Algorithms for Mining Inter-stream Associations from Large Sensor Networks. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 143–149. Springer, Heidelberg (2005)
9. Manku, G.S., Motwani, R.: Approximate frequency counts over data streams. In: The 28th International Conference on Very Large Data Bases, pp. 346–357 (2002)
10. Nagao, M.: Natural Language Processing, Iwanami (1996) (in Japanese)
11. Shimizu, K., Miura, T.: Disjunctive Sequential Patterns on Single Data Sequence and its Anti-Monotonicity. In: Perner, P., Imiya, A. (eds.) MLDM 2005. LNCS (LNAI), vol. 3587, pp. 376–383. Springer, Heidelberg (2005)
12. Shimizu, K., Miura, T.: Online Analysis for Disjunctive Sequential Patterns. In: ADBIS Workshop on Data Mining and Knowledge Discovery (ADMKD), pp. 61–72 (2006)
13. Takano, Y., Iwanuma, K., Nabeshima, H.: A Frequency Measure of Sequential Patterns on a Single Very-Large Data Sequence and its Anti-Monotonicity. In: proc. FIT, pp. 115–118 (2004) (in Japanese)
14. Uejima, H., CMiura, T., Shioya, I.: Estimating Timestamp From Incomplete News CorpusC proc.DEWS, 3C-o4 (2005) (in Japanese)

A New Dissimilarity Measure Between Trees by Decomposition of Unit-Cost Edit Distance

Hisashi Koga, Hiroaki Saito, Toshinori Watanabe, and Takanori Yokoyama

Graduate Schools of Information Systems, University of Electro-Communications, Tokyo 182-8585, Japan

Abstract. Tree edit distance is a conventional dissimilarity measure between labeled trees. However, tree edit distance including unit-cost edit distance contains the similarity of label and that of tree structure simultaneously. Therefore, even if the label similarity between two trees that share many nodes with the same label is high, the high label similarity is hard to be recognized from their tree edit distance when their tree sizes or shapes are quite different. To overcome this flaw, we propose a novel method that obtains a label dissimilarity measure and a structural dissimilarity measure separately by decomposing unit-cost edit distance.

1 Introduction

Tree is useful for expressing various objects such as semi-structured data and genes [1]. For this reason, it is essential to compute tree similarity in the field of pattern recognition and information retrieval.

In this paper we focus on labeled ordered trees with the root. Let T be a rooted tree. T is called a *labeled tree* if each node is a assigned a symbol from a finite alphabet Σ. T is *ordered* if a left to right order among siblings in T is given. Tree edit distance [2] is one of the most common dissimilarity measures between two trees and defined as the minimum cost necessary to convert from one tree to another tree by repeating node edit operations (i.e, deletion, insertion and relabeling). Tree edit distance is easily implemented with dynamic programming for labeled ordered trees [3]. To compute a tree edit distance, users need to supply a cost function defined on each edit operation. Because it is difficult to tailor node edit costs for a specific application, *unit-cost edit distance* [4] in which all of node edit operations cost 1 equally is used frequently.

Tree edit distance including unit-cost edit distance mixes the similarity of node labels and that of tree structure, because not only label of nodes but also tree shape are matched in turning a tree T_1 to another tree T_2. Thus, even if T_1 and T_2 share many nodes with the same label, the high label similarity is hard to be recognized from their tree edit distance, if their tree sizes or shapes are quite different. To overcome this flaw, this paper newly proposes to decompose unit-cost edit distance into node edit operations to match node labels and into those to match tree structure and, then, to obtain a label dissimilarity measure from the former and a structural dissimilarity measure from the latter. Since

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 643–652, 2007.

both measures take a value between 0 and 1, users can easily understand the extent of dissimilarity. Furthermore, our label dissimilarity measure generalizes tree inclusion [2] which is the problem to decide if a tree T_1 includes another tree T_2. That is, it can measure the extent of the tree inclusion, even if T_2 is not completely contained in T_1. Since our dissimilarity measures are obtained from unit-cost edit distance, users may use unit-cost edit distance without additional overhead like development of a new program, in case they are not satisfied with our dissimilarity measures, which is a large advantage of our approach.

By applying our method to the noisy subsequence tree recognition problem [5] and to the classification of XML documents, we show that our method yields a better performance than the unit-cost edit distance.

The structure of this paper is as follows. Sect. 2 introduces the unit-cost edit distance and the tree inclusion as preliminaries. Sect. 3 exemplifies the flaw of the unit-cost edit distance. Sect. 4 presents our dissimilarity measures. Sections 5 and 6 report the experimental results. Sect. 7 is the conclusion.

2 Preliminaries

2.1 Unit-Cost Edit Distance

Here we define unit-cost edit distance between trees. Let T_1 and T_2 be labeled ordered trees with the root. A node with a label x is denoted by "node x". T_1 can be converted to T_2 by repeating deletion, insertion and relabeling of nodes.

All of the insertion, deletion and relabeling of nodes are named *node edit operations*. These operations are defined formally as follows.

insertion: Let $u_1, u_2, \ldots, u_l$ be the children of node y that are ordered, where l is the number of children nodes for y. Inserting a node x as the child of y between u_i and u_j $(1 \leq i < j \leq l)$ means that x becomes a child of y and the parent of nodes from u_{i+1} to u_{j-1}.

deletion: Deleting a node x means that the children of x become the children of the parent of x and then x is removed.

relabeling: Relabeling a node x to a node y means that the label of the node is modified from x to y. It has no influence on the tree shape.

By introducing the notation of a null node λ, all of these operations can be consistently described in the form of a node pair (x, y), where (x, y) indicates that node x is changed to node y. A relabeling operation corresponds to the case when $x \neq \lambda$ and $y \neq \lambda$. If $x = \lambda$ and $y \neq \lambda$, (x, y) becomes an insertion operation. If $x \neq \lambda$ and $y = \lambda$, (x, y) grows a deletion operation.

When T_1 is converted to T_2, we denote this conversion by $T_1 \rightarrow T_2$. The tree conversion is determined uniquely by the set of the performed node edit operations. This set is expresses as $M(T_1, T_2)$ and M is called a *tree mapping*. Fig. 1 illustrates the tree mapping. Let $T[i]$ be the i-th node in T. A dotted line from node $T_1[i]$ to node $T_2[j]$ indicates that $T_1[i]$ is relabeled to $T_2[j]$. The nodes in T_1 not touched by a dotted line are deleted and those in T_2 are inserted.

A node edit operation (x, y) is associated with its cost $c(x, y)$. For computing unit-cost edit distance, we assume that $c(x, y) = 1$ for any x, y satisfying $x \neq y$

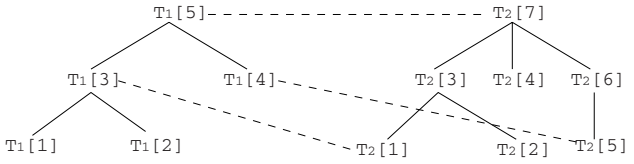


Fig. 1. A tree mapping

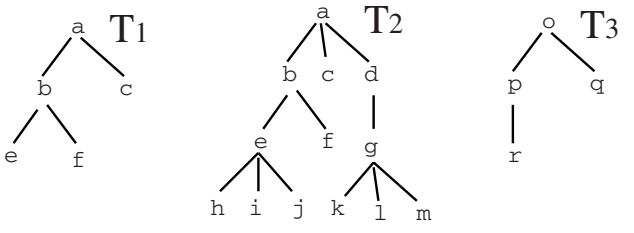


Fig. 2. 3 trees

and that $c(x, x) = 0$ for any x. This means that any insertion, deletion and relabeling to a different label costs 1 evenly. Relabeling to a different label is referred to as *non-free relabeling*.

Let D_M (I_M) be the set of nodes deleted from T_1 (respectively inserted to T_2) in M. Let S_M be the set of relabeled node pairs in M. Then, the cost of M is defined as (1) that is the total cost incurred in deletion, insertion and relabeling.

$$\mathrm{cost}(M) = \sum_{(v,w)\in S_M} c(v, w) + \sum_{v\in D_M} c(v, \lambda) + \sum_{w\in I_M} c(\lambda, w). \tag{1}$$

$D(T_1, T_2) = \min_M\{\mathrm{cost}(M)\}$ where the minimum is taken over M is called the unit-cost edit distance between T_1 and T_2.

2.2 Tree Inclusion

When T_1 can be converted to T_2 only by node delete operations, T_2 is said to be included in T_1. Tree inclusion problem is to determine if T_2 is included in T_1.

3 Flaw of Unit-Cost Edit Distance

Unit-cost edit distance contains the similarity of node labels and that of tree structure simultaneously. Hence, even if a pair of trees share many nodes with the same label, the high label similarity between them is hard to be recognized from the unit-cost edit distance, if their tree sizes or shapes are much different.

Fig. 2 illustrates the above claim. In this figure, T_1 is a subtree of T_2 and every node label that appears in T_1 also emerges in T_2. By contrast, T_3 do not include the same label as T_1 at all. Here, $D(T_1, T_2)$, the unit-cost edit distance between

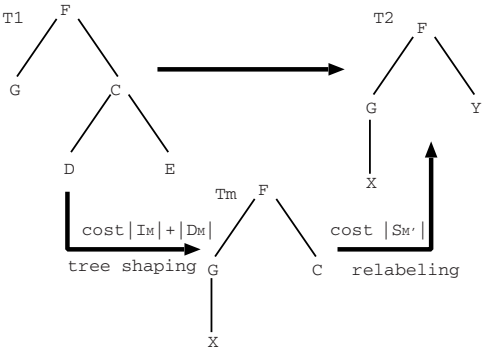


Fig. 3. Decomposition of tree conversion

T_1 and T_2, becomes 8, where the optimal tree mapping is to insert 8 nodes into T_1. On the other hand, $D(T_1, T_3) = 5$, where the optimal tree mapping is to delete node f from T_1 and relabel the remaining 4 nodes in T_1. Thus, T_1 becomes closer to T_3 than T_2 under unit-cost edit distance, despite the label similarity is higher between T_1 and T_2 than between T_1 and T_3.

This implies that unit-cost edit distance does not suit for applications for which label similarity should be paid much attention.

4 Our Dissimilarity Measures

After mentioning the decomposition of unit-cost edit distance in Sect. 4.1, our measures are defined in Sect. 4.2.

4.1 Decomposition of Unit-Cost Edit Distance

Especially, given two trees T_1 and T_2, we decompose the optimal tree mapping from a tree with more nodes to a tree with less nodes into node edit operations to match node labels and those to match tree structure. Let $|T_1| \geq |T_2|$ in the subsequence, where $|T|$ indicates the number of nodes in T and called the *tree size* of T. When $|T_1| < |T_2|$, they are permuted. Let $M(T_1, T_2)$ be the optimal tree mapping from T_1 to T_2 which corresponds to the unit-cost edit distance. We use Fig. 3 for explanation. Among the three types of node edit operations, only insertion and deletion are related to the change of the tree shape. Hence, we may suppose that M follows the next two steps in order.

Step 1: The tree shape is matched to T_2 by insertion and deletion operations.
Step 2: After Step 1, the node labels are made consistent with T_2 by relabeling.

Step 1 is further divided into two substeps in the next way.

Step 1a: The tree size is matched to T_2 by deleting some nodes from T_1.
Step 1b: The tree shape is matched between two trees of the same size by insertion and deletion operations.

Step 1 becomes $T_1 \rightarrow T_m$ and Step 2 becomes $T_m \rightarrow T_2$ in Fig. 3. Let S'_M is the set of non-free relabeling operations performed in M. Then the cost in Step 1 is $|I_M| + |D_M|$ and that in Step 2 is $|S'_M|$. In the example of Fig. 3, as two nodes D and E are removed and node X is inserted in Step 1, $|I_M| + |D_M| = 3$. Then, as Step 2 includes only one non-free relabeling operation (C, Y), $|S'_M| = 1$.

Since the cost in Step 1a is obviously $|T_1| - |T_2|$, the cost in Step 1b becomes $|I_M| + |D_M| - (|T_1| - |T_2|)$. Note that the number of insertion operations and that of deletion operations are the same in Step 1b, because the tree size does not change in Step 1b. In addition, insertion operations are performed only in Step 1b. Hence, we have $|I_M| + |D_M| - (|T_1| - |T_2|) = 2|I_M|$.

Because the nodes inserted in Step 1b (e.g. node X in Fig 3) must have labels included in T_2, the matching of labels is realized by $|I_M|$ insertion operations in Step 1b and $|S'_M|$ relabeling operations in Step 2. On the other hand, the matching of tree structure is realized by deletion operations in Step 1a and insertion and deletion operations in Step 1b.

4.2 Definitions of Our Dissimilarity Measures

Label Dissimilarity Measure: In the optimal tree mapping M, the number of node edit operations for matching labels is exactly $|I_M| + |S'_M|$. The label dissimilarity measure is defined as (2) in which $|I_M| + |S'_M|$ is normalized by the tree size $|T_2|$. The term "label dissimilarity measure" is abbreviated as LDM.

$$LDM(T_1, T_2) = \frac{|I_M| + |S'_M|}{|T_2|} \tag{2}$$

Since the matched labels remain in T_2, $0 \leq |I_M| + |S'_M| \leq |T_2|$. Hence, LDM takes a value between 0 and 1. Note that LDM takes the preservation of the order of nodes common to T_1 and T_2 into account. LDM has the next features.

- If and only if T_1 includes T_2, the LDM between them equals 0, because the optimal tree mapping consists of only deletion operations.
- When T_1 and T_2 do not have any common label at all, the LDM becomes 1, since any node in T_2 must be prepared by means of insertion or relabeling.

For the trees in Fig. 3, the LDM grows $\frac{1+1}{4} = 0.5$. In this way, LDM generalizes tree inclusion and measures the extent that T_1 includes T_2, even if T_2 is not completely included in T_1.

Structural Dissimilarity Measure (SDM) is defined as Formula (3).

$$SDM(T_1, T_2) = \frac{1}{2}\left(\frac{|T_1| - |T_2|}{|T_1|} + \frac{|I_M|}{|T_2|}\right) \tag{3}$$

The first term in (3) divides the cost for Step 1a by T_1 to exclude the influence of the tree size. Obviously, $0 \leq \frac{|T_1|-|T_2|}{|T_1|} \leq 1$. The second term in (3) corresponds to the cost of matching the shapes of the two trees of the same size in Step 1b, which equals $2|I_M|$. Note that the size of the two trees in Step 1b is $|T_2|$. As a

tree mapping which deletes all nodes from one tree and then inserts all nodes contained in T_2 is feasible, the cost for Step 1b is at most $2|T_2|$. Thus, we have $0 \leq \frac{|I_M|}{|T_2|} \leq 1$. Hence, SDM takes a value between 0 and 1.

4.3 Related Works

LDM works as a measure for approximate tree inclusion. With respect to approximate tree inclusion, Schlieder and Naumann [6] measures the quality of a tree inclusion by the number of nodes skipped in the tree mapping. Pinter et al. [7] allows the inexact matching of node labels in subtree homeomorphism, a special case of tree inclusion. They rank subtree homeomorphisms by label similarity. These two works disallow inexact tree inclusion in terms of tree structure unlike our approach. Sanz et al. [8] studies approximate subtree identification which admits an inexact matching of tree structure like our paper. Although their algorithm is very fast, it cannot recognize the exact tree inclusion, as the ancestor relationship is weakened. Bunke and Shearer [9] proposes a dissimilarity measure in which the size of the maximum common embedded subtree is divided by $\max\{|T_1|, |T_2|\}$. Their measure generalizes not tree inclusion but graph isomorphism.

5 Application to Noisy Subsequence Tree Recognition

Our LDM is especially suitable for the noisy subsequence tree recognition [5] that is formulated as follows:

Suppose we have a database DB of labeled ordered trees. Let T be any tree from DB. U_T is an arbitrary subtree of T obtained by randomly deleting nodes from T. A noisy subsequence tree Y_T of T is constructed by garbling U_T by insertion, deletion and relabeling further. Fig. 4 illustrates an example. Here, the nodes surrounded by a circle in T constitutes U_T. The nodes surrounded by a rectangle in U_T and Y_T correspond to noises in the tree conversion $U_T \rightarrow Y_T$. The task of the noisy subsequence tree recognition problem is to identify the original tree T from the trees in DB, given Y_T. One major application of this problem is the comparison of RNA secondary structures.

For this problem, Oommen and Loke [5] computed the constrained tree edit distances between Y_T and every tree in DB and judged the tree in DB that is the least dissimilar to Y_T as the original tree T. The constrained tree edit distance is a special tree edit distance under the condition that the number of relabeling operations executed in the tree mapping is fixed. In particular, they assume that the number of relabeling operations denoted by L executed in $U_T \rightarrow Y_T$ can be obtained by some means. The computational complexity of the constrained tree edit distance between two trees T_1 and T_2 becomes $O(|T_1||T_2| * \min\{|T_1|, |T_2|\}^2 * span(T_1) * span(T_2))$, where $span(T) = \min\{\text{No. of leaves in } T, \text{No. of depths in } T\}$.

Our method utilizes the LDM instead of the constrained tree edit distance. The computational complexity of LDM is $O(|T_1||T_2| * span(T_1) * span(T_2))$ which inherits from tree edit distance. Our method is splendid, as it does not need L.

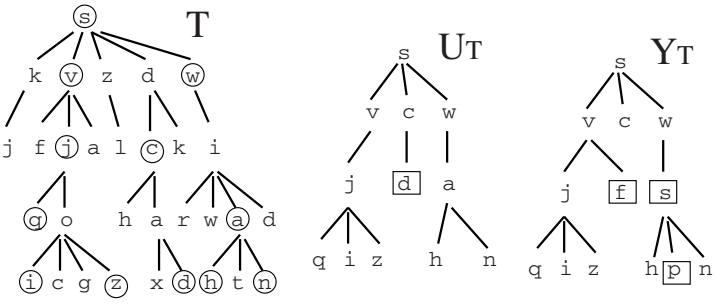


Fig. 4. Tree T, Subtree U_T and Noisy Subsequence Tree Y_T

Our method is compared with the one by Oommen and Loke. We perform the same experiment as their paper [5]: We prepare 25 labeled ordered tree as DB which vary in sizes from 25 to 35 nodes. A label of a node is chosen uniformly randomly from the English alphabet. For a tree T in DB, a corresponding noisy subsequence tree is constructed in the following manner.

1. 60% of the nodes in T are randomly selected and removed to produce U_T.
2. In making Y_T, each node in U_T is deleted with a probability of 5% and relabeled with a probability that follows the QWERTY confusion matrix in [5] which models the errors in stroking a keyboard. Also, several nodes are inserted to randomly chosen places in U_T such that the number of inserted nodes follows the geometric distribution with an expectation value of 2.

10 noisy subsequence trees are made per a tree in DB. Thus, 250 noisy subsequence trees are generated in total. The average number of noises to deform U_T is 3.67 that consists of 1.98 insertion, 0.53 deletion and 1.16 relabeling operations. The average size of 25 trees in DB is 30.7 and that of the 250 noisy subsequence trees is 13.8.

For each noisy subsequence subtree, its original tree is searched from DB both with our method and Oommen's method [5]. As the result, 99.6 % out of the 250 noisy sybsequence trees are correctly recognized by our method, which is superior to the success ratio of 92.8% by Oommen's method reported in [5]. Though we also implemented the unit-cost constrained tree edit distance, we could not attain a success ratio higher than 90%. The execution time of our method is 21.8s, whereas that of Oommen's method is 114.4s. Each execution time contains the time to compute a dissimilarity measure $250 \times 25 = 6250$ times. Our method is faster than Oommen's method, since unit-cost edit distance is lighter to compute than unit-cost constrained edit distance.

We remark here that even if 5 relabeling operations and 5 insertion operations are performed on each U_T to create each Y_T, the success ratio still grows about 98%. Roughly speaking, LDM is not affected by the gap of the tree sizes between T and U_T, so LDM is robust.

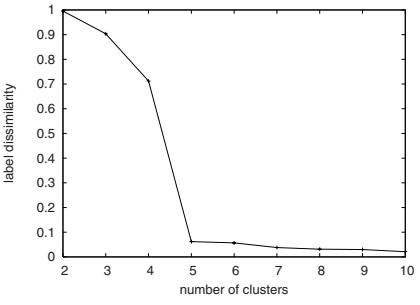


Fig. 5. Estimation of the number of clusters

6 Application to Classification of XML Documents

This section demonstrates that our measures yield a more natural clustering result (that is, a dendrogram) when combined with hierarchical clustering algorithms than the unit-cost edit distance in classifying XML documents. Especially, when a set of XML documents from multiple different XML databases are given, our measures are good at bundling the XML documents from the identical database into the same cluster. Our method works in two phases as follows.

Step 1: The hierarchical clustering is executed by using the LDM only. From the clustering result, clusters $C_1, C_2, \ldots, C_k$ are determined.
Step 2: The hierarchical clustering is performed once more. This time, we use a weighted sum $L_{ij} + \alpha S_{ij}$ as the dissimilarity measure between a tree i and a tree j. Here, L_{ij} and S_{ij} are the LDM and the SDM between tree i and tree j. The weighting parameter α is determined from the constraint that the membership of clusters $C_1, C_2, \ldots, C_k$ remains unchanged. As the result, α is not so large.

Our method aims to categorize the XML documents from the same database into the same cluster in Step 1, because LDM can equate them without regard to the number of repeatable tags or elements that are specified with the '*' regular expression in the schema. Note that these repeatable tags/elements cause tree structural difference among the XML documents from the same database. After Step 1, Step 2 attempts to classify the XML documents inside each cluster, considering their structural dissimilarities.

Step 1 need to determine the number of clusters k. k can be estimated from the dendrogram such that a sudden increase of the LDM value between a pair of clusters to be merged in the agglomeration signifies that two heterogeneous clusters that should not be united are merged. Step 2 produces a single dendrogram over the whole data by using a weighted sum of the LDM and the SDM.

6.1 Experimental Results

80 XML documents are sampled from the next 4 different XML databases (that is, 20 documents per a database): (1) XML-Actors, (2) the database of car

Table 1. Performance comparison to the unit-cost edit distance

	Actors	Car	IndexTerms	OrdinaryIssue	No. of Misses
Our Method	20	20	20	20	0
Unit-Cost Edit Distance	9	20	20	19	12

catalogs from Edmunds.com, (3) ACM SIGMOD RECORD IndexTermsPage and (4) ACM SIGMOD RECORD OrdinaryIssuePage.

These documents are classified with our method, where the group averaging method is adopted as a hierarchical clustering algorithm. Fig. 5 displays the LDM values of the merged clusters in Step 1. Since the LDM rises greatly when the number of clusters is reduced from 5 to 4, the number of clusters is determined as 4, which is the correct answer. Each of the 4 clusters contains exactly 20 XML documents that come from the same database. Table 1 compares our method with the unit-cost edit distance. It shows how many XML documents from the same database appear as a single clump on the dendrogram. Inferior to our method, the unit-cost edit distance fails for 12 XML documents because it is annoyed by the difference in tree sizes. The final dendrogram by our method after Step 2 is published on our web page [10].

Instead, the final dendrogram by our method for a smaller dataset is presented here. The dataset consists of 4 documents from the XML-actors (i.e., A1,A2,A3,A4), 4 documents from the IndexTermsPage (i.e., I1, I2, I3, I4) and 8 documents from the OrdinaryIssuePage (i.e., from O1 to O7). The dendrogram is described in Fig. 6 where $\alpha = 2.8$. The three clusters are separated clearly.

Table 2 shows the number of 'articlesTuple' elements in the XML documents from the OrdinaryIssuePage database. This element is defined as a repeatable element in the schema. The cluster for the OrdinaryIssuePage database classifies the members according to the number of the elements, reflecting the structural dissimilarity among the members. Since each articlesTuple element corresponds to a technical paper in one journal issue, our method is to categorize several journal issues according to the number of papers published in them.

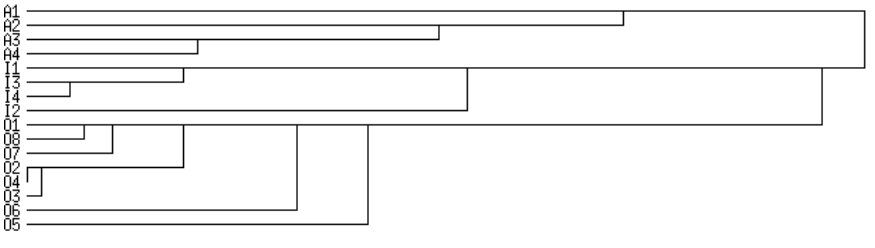


Fig. 6. Clustering result by our measures

Table 2. The number of the 'articlesTuple' elements in XML documents

XML document	O1	O2	O3	O4	O5	O6	O7	08
No. of appearances	8	7	7	7	4	10	9	8

7 Conclusion

This paper proposes a novel method to extract a label dissimilarity measure and a structural dissimilarity measure between two trees separately by decomposing their unit-cost edit distance. As our dissimilarity measures are derived from unit-cost edit distance with a little overhead, they are expected to complement unit-cost edit distance for applications for which unit-cost edit distance do not perform well for the reason that unit-cost edit distance mixes the similarity of node labels and that of tree structure. Furthermore, our label dissimilarity measure works as a measure for approximate tree inclusion and can evaluate the extent of tree inclusion, if a tree is not completely included in another tree. We verify the effectiveness of our dissimilarity measures with two experiments.

References

1. Moulton, V., Zuker, M., Steel, M., Pointon, R., Penny, D.: Metrics on RNA Secondary Structures. J. of Computational Biology 7, 277–292 (2000)
2. Bille, P.: A Survey on Tree Edit Distance and Related Problems. Theoretical Computer Science 337, 217–239 (2005)
3. Zhang, K., Shasha, D.: Simple Fast Algorithms for the Editing Distance between Trees and Related Problems. SIAM J. on Computing 18, 1245–1262 (1989)
4. Shasha, D., Zhang, K.: Fast Algorithms for the Unit Cost Editing Distance between Trees. J. of Algorithms 11, 581–621 (1990)
5. Oommen, B.J., Loke, R.K.S.: On the Pattern Recognition of Noisy Subsequence Trees. IEEE Trans. on PAMI 23(9), 929–946 (2001)
6. Schlieder, T., Naumann, F.: Approximate Tree Embedding for Querying XML Data. In: Proc. of ACM SIGIR Workshop on XML and Information Retrieval (2000)
7. Pinter, R.Y., Rokhlenko, O., Tsur, D., Ziv-Ukelson, M.: Approximate Labelled Subtree Homeomorphism. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) CPM 2004. LNCS, vol. 3109, pp. 59–73. Springer, Heidelberg (2004)
8. Sanz, I., Mesiti, M., Guerrini, G., Llavori, R.B.: Approximate Subtree Identification in Heterogeneous XML Documents Collections. In: Bressan, S., Ceri, S., Hunt, E., Ives, Z.G., Bellahsène, Z., Rys, M., Unland, R. (eds.) XSym 2005. LNCS, vol. 3671, pp. 192–206. Springer, Heidelberg (2005)
9. Bunke, H., Shearer, K.: A Graph Distance Metric based on the Maximal Common Subgraph. Pattern Recognition Letters 19, 255–259 (1998)
10. http://sd.is.uec.ac.jp/~koga/IDEALdata.html

Optimizing Web Structures Using Web Mining Techniques

Jonathan Jeffrey[1], Peter Karski[1], Björn Lohrmann[1], Keivan Kianmehr[1], and Reda Alhajj[1,2]

[1] Computer Science Dept, University of Calgary, Calgary, Alberta, Canada
[2] Department of Computer Science, Global University, Beirut, Lebanon
alhajj@ucalgary.ca

Abstract. With vibrant and rapidly growing web, website complexity is constantly increasing, making it more difficult for users to quickly locate the information they are looking for. This, on the other hand, becomes more and more important due to the widespread reliance on the many services available on the Internet nowadays. Web mining techniques have been successfully used for quite some time, for example in search engines like Google, to facilitate retrieval of relevant information. This paper takes a different approach, as we believe that not only search engines can facilitate the task of finding the information one is looking for, but also an optimization of a website's internal structure, which is based on previously recorded user behavior. In this paper, we will present a novel approach to identifying problematic structures in websites. This method compares user behavior, derived via web log mining techniques, to an analysis of the website's link structure obtained by applying the Weighted PageRank algorithm (see [19]). We will then show how to use these intermediate results in order to point out problematic website structures to the website owner.

Keywords: data mining, web mining, user behavior, search engines, PageRank.

1 Introduction

Website complexity is constantly increasing, making it more difficult for users to quickly locate the information they are looking for. This, on the other hand, becomes more and more important due to the widespread reliance on the many services available on the Internet nowadays. Analyzing the user's behavior inside the website structure, will provide insight on how to optimize the website's structure to improve usability.

Web Mining is the use of data mining methods to identify patterns and relationships amongst web resources. It is basically classified into web mining: web content, web usage and web structure mining; the last two are used to solve the website structure optimization problem. Web structure mining involves the crawling and analysis of web page content to identify all links existing within

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 653–662, 2007.

the page, which will then be used to create a directed graph representing the structure of the site being mined. Each node within this graph signifies an individual page and each edge is a link between two pages. On the other hand, web usage mining requires the parsing of web server logs to identify individual user behavior. Specifically, the sites visited, total visits and total time spent looking at the page, also known as "think time", are considered. These values are parsed from original server logs, or could be taken from preprocessed logs as well. Furthermore, in this paper, we explain how to use the Weighted PageRank algorithm [2,3,19,20] for web-structure mining to analyze the hyperlink structure of a website. Also, we demonstrate how to use web log mining to obtain data on the site user's specific navigational behavior. We then describe a scheme how to interpret and compare these intermediate results to measure the website's efficiency in terms of usability. Based on this, it shall be outlined how to make recommendations to website owners in order to assist them in improving their site's usability.

The rest of this paper is organized as follows. Section 2 is related work. Section 3 describes the proposed approach; we first present how web structure mining is utilized in the process of website optimization; then describe the participation of web usage mining to the process; then we discuss how the overall recommendation is conveyed to the user of the analyzed website. Section 4 reports test results that demonstrate the applicability and effectiveness of the proposed approach. Section 5 is summary and conclusions.

2 Related Work

As described in the literature, numerous approaches have been taken to analyze a website's structure and correlate these results with usability, e.g., [6,7,9,10,11,15,16]. For instance, the work described in [14] devised a spatial frequent itemset data mining algorithm to efficiently extract navigational structure from the hyperlink structure of a website. Navigational structure is defined as a set of links commonly shared by most of the pages in a website. The approach is based on a general purpose frequent itemset data mining algorithm, namely ECLAT [5]. ECLAT is used to mine only the hyperlinks inside a window with adaptive size, that slides along the diagonal of the website's adjacency matrix. They compared the results of their algorithm with results from a user-based usability evaluation. The evaluation method gave certain tasks to a user (like for example finding a specific piece of information on a website) and recorded the time needed to accomplish a task and failure ratios. The researchers found a correlation between the size of the navigational structure set and the overall usability of a website, specifically the more navigational structure a website has, the more usable it is as a general rule of thumb.

In [18], it is proposed to analyze the web log using data mining techniques to extract rules and predict which pages users will be going to be based on their prior behavior. It is then shown how to use this information to improve the website structure. By its use of data mining techniques, this approach is related

to our approach described in this paper, although the details of the method vary greatly, due to their use of frequent itemset data mining algorithms. The main difference between our approach and the method described in [18] is that they do not consider the time spent on a page by a visitor in order to measure the importance of that particular page. Their approach applies frequent itemset mining that discovers navigation preferences of the visitors based on the most frequent visited pages and the frequent navigational visiting patterns. However, we believe that in a particular frequent navigational pattern there might exist some pages which form an intermediate step on the way to the desirable page that a user is actually interested in. Therefore, the time spent on a page by a visitor has to be considered as an important measure to quantify the significancy of a page in a website structure.

The work described in [13] proposed two hyperlink analysis-based algorithms to find relevant pages for a given Web page. The work is different in nature from our work; however it applies web mining techniques. The first algorithm extends the citation analysis to web page hyperlink analysis. The citation analysis was first developed to classify core sets of articles, authors, or journals to different fields of study. In the context of the Web mining, the hyperlinks are considered as citations among the pages. The second algorithm makes use of linear algebra theories to extract more precise relationships among the Web pages to discover relevant pages. By using linear algebra, they integrate the topologic relationships among the pages into the process to identify deeper relations among pages for finding the relevant pages. The work described in [12] describes an expanded neighborhood of pages with the target to include more potentially relevant pages.

In the approach described in [19], the standard PageRank algorithm was modified by distributing rank amongst related pages with respect to their weighted importance, rather than treating all pages equally. This results in a more accurate representation of the importance of all pages within a website. We used the Weighted PageRank formula outlined in [19] to complement the web structure mining portion of our approach, with the hope of returning more accurate results than the standard PageRank algorithm.

In [21], the authors outline a method of preparing web logs for mining specific data on a per session basis. This way, an individual's browsing behavior can be recorded using the time and page data gathered. Preparations to the log file such as stripping entries left by robots are also discussed.

3 Overview

In order to achieve our goal of recommending changes to the link structure of a website, we have identified two main subproblems which must be initially solved. First, to determine which pages are important, as implied by the structure of the website. Second, to conclude which pages the users of this website consider to be important, based on the information amassed from the web log. Once we have solved these two subproblems, we now have methods in place which give us two different rankings of the same web pages. Our final task is implementing

a scheme to compare the results of the first two problems and make meaningful recommendations. In the subsections which follow, we will discuss the algorithms we will use for unraveling each of these tasks and the reasoning behind these algorithms.

3.1 Web Structure Mining

Web structure mining involves crawling through a series of related web pages (for example all pages inside a user defined subdomain), extracting meaningful data that identifies the page and use that data to give the page a rank based on given criteria. To begin with, a set or root of web pages is provided, an application called a crawler will traverse these pages and extract the needed information from them. The information we are interested in are the hyperlinks contained within the page.

Extracting this information can be done using regular expressions. There are challenges using regular expressions, because they assume that the code used within the web page follows all standards. Simple errors, such as some HTML tags not being closed or improperly formatted and non-HTML code such as CSS or javascript can throw off the parsing of the page and lead to inaccurate results.

Once the hyperlinks within the webpage have been extracted, the crawler will recursively continue crawling the web pages whose links were found in the current page after replicates have been removed, since crawling the same page twice is unnecessary. Any duplicate hyperlinks within the page will need to be removed, as well as any links that have already been processed or are already in the queue awaiting processing. After having crawled the complete website or a user defined part of it, depending on what the user specifies, the standard page rank $PR(p_i)$ of each page p_i can be computed as

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)} \tag{1}$$

where N is the total number of pages that have been crawled, $M(p_i)$ is the set of pages that link to p_i, $L(p_j)$ is the number of outgoing links on page p_j, and d is a damping factor, usually chosen around 0.85. The damping factor can be interpreted as the probability that a user follows the links on a page. It has been included due to the following observation: *Sometimes a user does not follow the links on a page p_k and just chooses to see a random page p_l by entering its address directly.* This should be considered when computing the page rank of p_l. Thus, in the above formula, $\frac{1-d}{N}$ can be seen as the influence of a random jump to page p_i on the page rank $PR(p_i)$.

In [19], an improved version of standard page rank is proposed. The weighted page rank algorithm (WPR) considers the fact that the page rank of a popular page should have a higher weight than the one of an unpopular page. The WPR value is computed as:

$$WPR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} PR(p_j) W^{in}_{(p_j,p_i)} W^{out}_{(p_j,p_i)} \tag{2}$$

Here, $W^{in}_{(p_j,p_i)}$ and $W^{out}_{(p_j,p_i)}$ are the weights of the link between documents p_j and p_i. They can be computed as:

$$W^{in}_{(p_j,p_i)} = \frac{I_{p_i}}{\sum_{p \in R(p_j)} I_p} \tag{3}$$

$$W^{out}_{(p_j,p_i)} = \frac{O_{p_i}}{\sum_{p \in R(p_j)} O_p} \tag{4}$$

where I_x is the number if links pointing to page x, O_x is the number of outgoing links in page x and $R(x)$ is the set of pages that are linked to from page x. Each element of the sum of page ranks is multiplied by it's respective weight. The result is that more important pages are given a higher page rank, unlike the original page rank algorithm that divides the rank of a page evenly amongst documents it links to. At the end, a more accurate result is achieved.

We chose to use WPR instead of the standard page rank, since it has proven to yield slightly better results in experiments (see [19]). The output of this processing stage is a list of $[p_i, WPR(p_i)]$ pairs, sorted in descending order by the WPR values.

3.2 Web Log Mining

We have decided to base the rankings which users give to web pages on two parameters: 1) frequency (number of visits), and 2) time (total time spent by all users at a web page).

Preprocessing: We need to take steps to "clean" the web log to minimize interference from robots before using it to generate the actual output of this stage. One approach we consider useful for this has been proposed in [21], where it is proposed to discard those sessions that match the following access patternts that are likely to be robots characteristics:

- Visiting around midnight, during light traffic load periods in order to avoid time latency.
- Using `HEAD` instead of `GET` as the access method to verify the validity of a hyperlink; the Head method performs faster in this case as it does not retrieve the web document itself.
- Doing breadth search rather than depth search; robots do not navigate down to low-level pages because they do not need to access detailed and specific topics
- Ignoring graphical content; robots are not interested in images and graphic files because their goal is to retrieve information and possibly to create and update their databases.

Based on the cleaned logfile, we then identify sessions, which in turn are used to compute the total number of visits v_i and the total time spent by users t_i for each page. It shall be noted that we must ensure that the number of user sessions we extract from the web log are of sufficient size to give us a realistic ranking of popular pages.

Computing log rank values: As already mentioned, the first parameter to consider in the process is the number of times a particular page was visited by our group of users. The fact that a user has visited a page may lead us to believe that they consider it an important page. While this is true in many cases, there is also the possibility that the page was just an intermediate step or "hop" on the way to the page which the user is actually interested in. A very high number of visits from page $A \rightarrow B$ and a relatively low total time spent at page B seem to imply that the page B is used primarily (or even exclusively) as a "hopping" point. In this case, a viable recommendation may be to change the link structure of the website so that the user is able to navigate directly from $A \rightarrow C$, without having to make a stop in-between at page B. Therefore, we need a way to give lesser weight to the visit counter and a higher weight to the total time in our ranking scheme.

Assuming that v_i is the number of visitors for a page i and t_i is the total time spent by all visitors on this page, the *log rank* value l_i shall be defined as:

$$l_i = 0.4v_i + 0.6t_i \tag{5}$$

Taking a weighted sum of the visits and time will result in a value that represents the importance of a page relative to the others. Pages that are frequently visited and accessed for long periods of time will have a larger log rank than pages with an insignificant number of visits and think time. Rather than giving time and visits equal importance as discussed above, the difference is quantified through a constant, in this case being a 60/40 split, respectively. Depending on the content of the web site being analyzed, these constants can be changed to account for the specific audience or purpose of the site. For example, a website with pages normally filled with large amounts of visual/textual information will have on average longer think times than pages sparsely filled with content. A balance between having fewer pages with large amounts of content, versus many pages with little content must be identified and quantified in the log rank function in order to be accurate. Finally, the output of this stage, is a list of pages sorted by their log rank value l_i.

3.3 Analysis

This is the last stage of processing and yields results directly for the users, in form of recommendations on how to change their website. The required input is:

- A list of pages with their page rank values $p_i \in R^+$, which is sorted by the page rank values.
- A list of websites with their ranking values $l_i \in R^+$ from the weblog mining, which is sorted by the ranking values.

Preprocessing: First, we need to preprocess the page rank and log ranking values. This step consists of normalizing them to a common index of integers. This is done by taking the list of page ranks and sorting them in descending

order. We then chose to assign an index to each page rank value, the largest page rank receiving index zero, and the smallest page rank receiving the highest index value. In the event that several pages share the same page rank value, they would receive the same index value. The same process is then repeated for the log rank values.

This step changes the log and page rank values from two different distributions into a simple linear distribution, which facilitates appropriate comparison. We validated this step during the evaluation of our method (see Section 4), where the log and page rank values had significantly different distributions, which yielded non-sensical analysis results.

The output of this step consists of two lists, the page rank and log rank indexes along with their respective page. At this step, the lists are still independent of each other.

Performing Analysis: After the preprocessing step, we can now compute the following value d_i for each page:

$$d_i = index(l_i) - index(p_i) \tag{6}$$

This calculates the difference in rank between the two rank values. Ideally, we will find $d_i = 0$, because there should be little deviation in the ranking of the page rank and log rank values. Finally, the pages will be sorted in ascending order according to their d_i values. At the top and the bottom of the list, we can distinguish the following cases:

1. The page has got a high page rank index and a low log rank index (d_i very low).
2. The page has got a low page rank index and a high log rank index (d_i very high).

In case 1, the software performing the analysis should recommend the user to put the site into a place, where it is harder to reach, in favor of pages that might require to be reachable more easily. This includes but is not limited to:

- Removing links to that page, especially on those pages with high page rank.
- Linking to the page from places with low page rank value instead.

In case 2, the software should recommend modifying the link structure in a fashion that makes the page easier to reach. This means, for example, adding links to that page, especially on suitable pages with high page rank.

The intuition for a very high or very low deviation generally being undesirable, is the following: One could interpret a high page rank value as a site being easily reachable from other (important) pages, whereas a low page rank value thus could be interpreted as an indicator, that the page is hard to reach. On the other hand, a high log rank value testifies that a page is popular, whereas low log rank values indicate unpopular pages. Therefore, in the first case with a very low d_i, the site is easy to reach, but only few people actually want to see it; and in the second case with a very high d_i, the page is very hard to reach for visitors, but comparably many people want the information on it and have to spend time looking for it. Thus, it is natural, that according to this scheme, an ideally positioned page has a value $d_i \approx 0$.

The Relinking Process: The aforementioned "relinking" process has to be carried out manually by the website owner since he/she has to consider the content structure of the page; thus it is out of the scope of this algorithm to propose concrete relinking in terms of "Link page A to B" or "Remove the link on page A". The outlined algorithm merely represents a support in determining possibly misplaced pages and in deciding where to add or remove links (page rank values can be helpful here).

To assist him/her in the process, after this stage, the website owner should be presented with the following information:

- The sorted list of pages (called *UNLINK* list), with $d_i < 0$ and $d_i^* > \epsilon_1$, where ϵ_1 is a user defined threshold.
- The sorted list of pages (called *LINK-TO* list), with $d_i > 0$ and $d_i^* > \epsilon_2$, where ϵ_2 is another user defined threshold.
- For each webpage in the above lists, provide the set of pages that link to it (incoming links) and the set of pages that are being linked to from it (outgoing links).
- The page rank and log rank value for each webpage that has been analyzed, including but not limited to those in the *UNLINK* and the *LINK-TO* list.

This information should be sufficient to detect and resolve design issues in a website's structure that affect usability. The ranking approach is supportive in that it helps the owner to focus on the important issues. To guide the process of relinking or altering the structure, page rank and log rank values are provided.

4 Evaluation

We tested our algorithm on a medium sized website ($\approx$ 631 pages) obtained from [17], which provides reference for HiFi devices. Its structure is mostly wider than deep, as for example when it lists the manufacturers of documented devices. Since this website has been provided for experiments with data mining techniques, it already came with a log file that had been parsed into sessions. Performing the analysis on the site yielded the distribution of deviation values d_i as shown in Figure 1.

As can be seen from Figure 1, we have a relatively low number of pages with a deviation far from the ideal value. The majority of the pages fall within a small margin of ± 200, which is still acceptable. Some pages like for example /dr-660/index.html (lowest d_i value) showed a large discrepancy between user popularity and reachability, since it was linked to from one of the central pages, but hardly received any hits. Other pages like /manufacturers/korg/s-3/index.html (second highest d_i value) appear to have been very popular with the site users, but are relatively hard to reach since they are hidden "deep" in the website's structure. A viable change in this case would be to provide a link to it on the pages at or close to the website's document root (for example in a "Favorites" or "Recommendations" section), since this is where the users start browsing. Further investigation of the highest and lowest values, showed

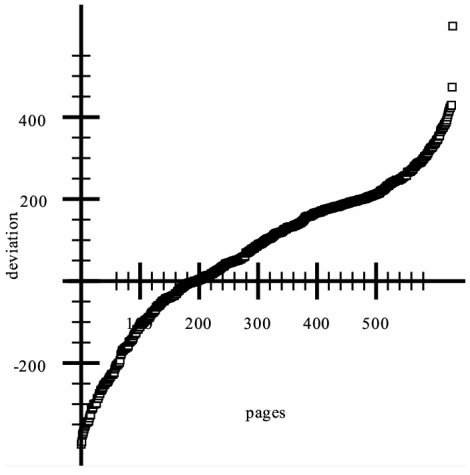


Fig. 1. Plot of d_i (sorted by d_i)

the same tendency and thus revealed locations where relinking seemed necessary after manual investigation from our side.

Despite a certain "noise" (meaning pages that are classified as misplaced, but cannot be really relinked), our method has succeeded to identify problematic locations in the website's structure.

5 Summary and Conclusions

In this paper, we explained how to use the Weighted PageRank algorithm for web-structure mining to analyze the hyperlink structure of a website. Further, we demonstrated how to use web log mining to obtain data on the site user's specific navigational behavior. Our approach then showed how to combine these values in order to measure a website's usability. We successfully validated our method using the data set provided under [17], which shows that this is a simple but viable approach to solve the given problem. In our opinion, a similar method should be used as part of a larger set of tools, when it comes to usability optimization of websites.

References

1. Abiteboul, S., Preda, M., Cobena, G.: Adaptive on-line page importance computation. In: Proc. of the International Conference on World Wide Web, pp. 280–290 (2003)
2. Altman, A., Tennenholtz, M.: Ranking systems: the pagerank axioms. In: Proc. of ACM Conference International on Electronic commerce, pp. 1–8 (2005)
3. Bianchini, M., Gori, M., Scarselli, F.: Inside pagerank. ACM Transactions on Internet Technology 5(1), 92–128 (2005)

4. Boldi, P., Santini, M., Vigna, S.: Pagerank as a function of the damping factor. In: Proc. of the International Conference on World Wide Web, pp. 557–566 (2005)
5. Borgelt, C.: Efficient implementations of apriori and eclat (November 14, 2003)
6. Borodin, A., Roberts, G.O., Rosenthal, J.S., Tsaparas, P.: Link analysis ranking: algorithms, theory, and experiments. ACM Transactions on Internet Technology 5(1), 231–297 (2005)
7. Bradley, J.T., de Jager, D.V., Knottenbelt, W.J., Trifunovic, A.: Hypergraph partitioning for faster parallel pagerank computation. In: Bravetti, M., Kloul, L., Zavattaro, G. (eds.) EPEW 2005. LNCS, vol. 3670, pp. 155–171. Springer, Heidelberg (2005)
8. Chakrabarti, S., Dom, B., Gibson, D., Kleinberg, J., Raghavan, P., Rajagopalan, S.: Automatic resource compilation by analyzing hyperlink structure and associated text. In: Proc. of the International Conference on World Wide Web (1998)
9. Chen, Y.-Y., Gan, Q., Suel, T.: I/o-efficient techniques for computing pagerank. In: Proc. of ACM International Conference on Information and knowledge management, pp. 549–557 (2002)
10. Chirita, P.-A., Diederich, J., Nejdl, W.: Mailrank: using ranking for spam detection. In: Proc. of ACM International Conference on Information and knowledge management, pp. 373–380 (2005)
11. Cho, J., Roy, S., Adams, R.E.: Page quality: in search of an unbiased web ranking. In: Proc. of ACM SIGMOD, pp. 551–562 (2005)
12. Dean, J., Henzinger, M.: Finding related pages in the world wide web. In: Proc. of the International Conference on World Wide Web (1999)
13. Hou, J., Zhang, Y.: Effectively finding relevant web pages from linkage information. IEEE Transactions on Knowledge and Data Engineering 15(4), 940–951 (2003)
14. Li, C.H., Chui, C.K.: Web structure mining for usability analysis. In: Proc. of IEEE/WIC/ACM International Conference on Web Intelligence, pp. 309–312 (2005)
15. Jiang, X.-M., Xue, G.-R., Song, W.-G., Zeng, H.-J., Chen, Z., Ma, W.-Y.: Exploiting pagerank at different block level. In: Proc. of the International Conference on Web Information Systems Engineering, pp. 241–252 (2004)
16. Massa, P., Hayes, C.: Page-rerank: Using trusted links to re-rank authority. In: Proc. of IEEE/WIC/ACM International Conference on Web Intelligence, pp. 614–617 (2005)
17. U. of Washington Artificial Intelligence Research. Music machines website, http://www.cs.washington.edu/ai/adaptive-data/
18. Renáta Iváncsy, I.V.: Frequent pattern mining in web log data. Journal of Applied Sciences at Budapest Tech 3(1), 77–90 (2006)
19. Xing, W., Ghorbani, A.A.: Weighted pagerank algorithm. In: Proc. of Annual Conference on Communication Networks and Services Research, pp. 305–314. IEEE Computer Society, Los Alamitos (2004)
20. Xing, W., Ghorbani, A.A.: Weighted pagerank algorithm. In: Proc. of Annual Conference on Communication Networks and Services Research, pp. 305–314 (2004)
21. Yu, J.X., Ou, Y., Zhang, C., Zhang, S.: Identifying interesting customers through web log classification. IEEE Intelligent Systems 20(3), 55–59 (2005)

A Collaborative Recommender System Based on Asymmetric User Similarity

Marta Millan, Maria Trujillo, and Edward Ortiz

School of Systems and Computer Engineering of Universidad del Valle,
Ciudad Universitaria Melendez, Cali, Colombia
{millan, mtrujillo}@univalle.edu.co

Abstract. Recommender systems could be seen as an application of a data mining process in which data collection, pre-processing, building user profiles and evaluation phases are performed in order to deliver personalised recommendations. Collaborative filtering systems rely on user-to-user similarities using standard similarity measures. The symmetry of most standard similarity measures makes it difficult to differentiate users' patterns based on their historical behaviour. That means, they are not able to distinguish between two users when one user' behaviour is quite similar to the other but not *vice versa*. We have found that the *k*-nearest neighbour algorithm may generate groups which are not necessarily homogenous. In this paper, we use an asymmetric similarity measure in order to distinguish users' patterns. Recommendations are delivered based on the users' historical behaviour closest to a target user. Preliminary experimental results have shown that the similarity measure used is a powerful tool for differentiating users' patterns.

Keywords: Recommender systems, collaborative filtering, asymmetric similarity measure, *k*-nearest neighbours.

1 Introduction

Recommender systems could be seen as an application of a data mining process [21] in which data collection, pre-processing, building user profiles and evaluation phases are performed in order to deliver personalised recommendations. The goal of recommender systems is to provide a user with personalized recommendations based on either his/her tastes and preferences or based on a group of people with similar tastes and preferences [1]. Five classes of recommendation techniques are proposed in terms of the background data, input data and the algorithm to generate recommendations: collaborative, content-based, demographic, utility-based and knowledge-based [5].

Collaborative filtering techniques, in particular, rely on user-to-user similarities but have three major limitations: sparsity, scalability, and cold-star. Several methods have been proposed for solving these limitations based on clustering and machine learning techniques [3][4][14][16][19]. ClustKNN [23] addresses the scalability problem applying the *k*-means algorithm for building a user model and the *k*-nearest neighbour

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 663–672, 2007.

(KNN) clustering algorithm for calculating predictions. Kim *et al.* [16] propose a probabilistic model generated under the assumption that items are not related to each other in any way, *i.e.* they are independent. A smoothing-based method is introduced under a hybrid collaborative filtering approach in [34]. From training data, initial clusters are calculated using the *k*-means algorithm. The Pearson correlation coefficient is used as a similarity measure function. Smoothing strategies are applied to unrated items.

In Godoy *et al.* [11] user-profiling approaches to develop agents that help users in finding, filtering and getting information tasks are reviewed. These approaches need data and information about users in order to capture user's profiles. To construct user's profiles it is necessary to infer information based on the user-system interaction. User profiles are based on knowledge acquired implicitly or explicitly from this interaction. Some sources of this information, proposed in Godoy *et al*. [11], include historical user navigation, access logs and relevance feedback – explicit or implicit. Explicit feedback requires that users assign values to an examinee item. Instead, implicit feedback is inferred based on implicit interest indicators.

In McLaughlin and Herlocker [19], a user nearest-neighbour (NN) algorithm is analysed and a belief distribution algorithm is introduced in order to improve user modelling. In this proposal, a predictive algorithm, solves two main drawbacks of NN approaches: few neighbours who have rated an item are available for a target user; neighbours with a very low correlation score to target user rated an item. The Pearson correlation is used to calculate the most similar N-users for a target user from account historical item rating data. Taking into account that user's rating are subjective a belief difference distribution is introduced from calculating correlations.

In [31], an approach of collaborative filtering was introduced in which user neighbourhood is calculated based on demographic data, psychographic data and users' historical behaviour. A weight similarity measure is proposed for clustering users in order to take into account dynamic human being behaviour. This measure is characterized by providing a way to define which characteristics are more important at a specific moment. Moreover, characteristics are used in a nominal scale of measurement since users' behaviour has no order when time is not taken into account.

We have found that the *k*-nearest neighbour (KNN) algorithm may generate groups which are not necessarily homogenous. This is probably due to the size of the available data. We have detected that these differences may underestimate the similarity between users.

In this paper, a collaborative recommender system based on an asymmetric measure is introduced. It is derived from the need to distinguish between two users when one user' behaviour is quite similar to other but not *vice versa*. We use an asymmetric similarity measure for distinguishing users' patterns [6][7]. In this approach, a user-to-user similarity matrix is built and clusters are extracted through thresholding. Recommendations are delivered based on the users' historical behaviour closest to a target user. Preliminary experimental results have shown that the similarity measure is a powerful tool for differentiating users' patterns.

The paper is organized as follows. Collaborative recommender systems basic concepts are described in Section 2. In Section 3, we introduce a recommender system based on asymmetric users' patterns. A digital library experimental framework where

our approach has been implemented is described in Section 4. Preliminary evaluation and final remarks are presented in Sections 5 and 6, respectively.

2 Collaborative Recommender Systems

In a collaborative filtering process, there is a set of m users, U= $\{u_1, u_2, \ldots, u_m\}$ and a set of n items, I=$\{i_1, i_2, \ldots, i_n\}$. To each user u_j, a list of items $L_j = \{i_k: 1 \leq k \leq n_j.\}$ is associated. L_j contains items on which a user has explicitly shown interest by assigning a rating score to them or implicitly, based on user behaviour. Collaborative filtering algorithms are applied in order to find interesting items for a target user. Interesting items can be obtained in two ways: prediction and recommendation [31]. In the prediction way, calculating a predictive score $P_{score}(i_q,u_j)$ that represents the level of interest that an item $i_q \notin L_j$ may have for user u_j. In the recommendation way, a list L of N-top items, such that $L \subset I$ and $L \cap L_j = \varnothing$, is delivered to user u_j. Thus, L contains items that could be interesting for user u_j.

Collaborative filtering user-based recommendation approaches try to identify neighbourhoods for each user based on similar features – *e.g.* demographics, psychographics, behavioural – [13][17][22][24][27]. Most collaborative techniques work based on ratings about items provided for users.

Collaborative filtering techniques can be classified as either model-based or memory-based [1]. The former builds a model from historical data to recommend other items [3] [4] [14] [25]. The latter uses a utility function in order to calculate similarity between users to build neighbourhoods [9][12] [15] [17][24][27] [32][33]. If a user is included in a neighbourhood – similar tastes are shared – it is possible to predict the utility of an item to him/her based on others items rated by users in the same neighbourhood.

Similarity measures are commonly evaluated using ratings as though they were quantitative values regardless of the fact that these measures are not defined for attributes or subjective user evaluations. The Pearson correlation and the cosine distance are often used to assess similarity between users [12] [20] [29] [32]. However, – according to Choi *et al.* [9] – these similarity measures have some weaknesses: scalability problems, applying limitations depending on the domain, and assuming that attributes are mutually independent. Garden *et al.* [10] and Herlocker *et al.* [12] have considered the use of the Spearman correlation, which is just the Pearson correlation coefficient computed for the ranks of the ratings.

3 A Recommender System Based on Asymmetric User Similarity

The proposed approach is integrated into an experimental computer sciences digital library. In this context, users are students and items are digital documents, such as papers, books, research reports and theses. This approach uses an unobtrusive method for recommendation calculations that takes into account information related to the documents he/she has previously downloaded.

The approach combines advantages of memory-based and model-based collaborative recommendation systems. Memory-based – in an off-line phase – deals

with user neighbourhood calculations in order to reduce the computational complexity – scalability problem. Model-based – in an online phase – uses a probability model for the preferences prediction based on user neighbourhood information. Thus, a user preferences pattern is calculated in an off-line phase. His/her preferences prediction is calculated in an on-line phase.

3.1 Asymmetric Users' Similarity

Collaborative recommender systems are based on an important feature of human behaviour which is the tendency to consume a limited set of items. Thus, a set of items becomes fashionable for a group of people with similar preferences. Collaborative filtering systems are based on similarity of users in which a neighbourhood to each user is generated. A neighbourhood is built taking into account user's similarity.

Although users hardly ever give explicit feedback and user interests change over time [26], most collaborative techniques work based on ratings about items provided for users. Ratings can be obtained explicitly or implicitly. Explicit ratings are subjective user evaluations or voting. The similarity between two users is evaluated using ratings as numeric values. Similarity measures – *e.g.* Pearson correlation coefficient and cosine distance – are applied regardless of the fact that these functions are not defined for subjective user evaluations. That is, the average of "very satisfied", "somewhat dissatisfied" and "very dissatisfied" does not exist [8] [28].

The normalised rank transformation, the Spearman correlation, the Kendall correlation, the Pearson correlation and the Footrule, the Cayley, the Hamming, the Ulam, the Chebyshev/Maximum value and the Minkowski distances are commonly used to calculate similarity or distance between two ranks or two rating vectors [30]. These measures are not asymmetric, that is, they can not capture differences between two users when one user has a lot more historical information than the other. The users' similarity measure has to distinguish when a user u_b has a lot more historical information than user u_a whilst user u_a has a quite similar historical information as user u_b. This could be seen as quantifying the similarity between u_a and u_b when $L_a \subseteq L_b$. In this case, the similarity score between user u_a and user u_b has to be large whilst the similarity score between user u_b and user u_a has to be small.

Similarity between users is defined as either exact or approximate historical information matches or coincidences of patterns. Thus, the similarity between u_a and u_b could be measured as follows [6] [7]:

$$S(u_a, u_b) = \frac{|L_a \cap L_b|}{|L_a|}, \tag{1}$$

where $|\cdot|$ is the cardinality of a set or a list of items.

By asymmetry, the similarity between u_b and u_a is given by:

$$S(u_b, u_a) = \frac{|L_a \cap L_b|}{|L_b|}. \tag{2}$$

A user-to-user similarity matrix is built using the similarity measure. In this matrix, the *j-th* row represents the similarity between user *j* and the set of users.

For instance, Figure 1 represents a set of users and their lists of downloaded data.

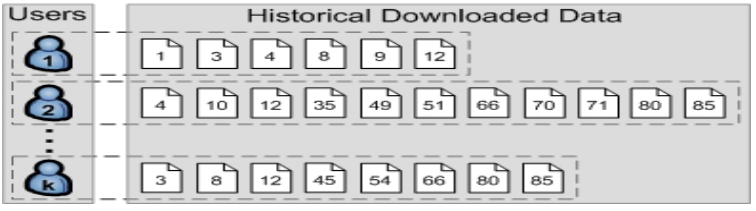


Fig. 1. Illustration of downloaded data

Thus, the user-to-user similarity matrix has to be fully calculated. A threshold is used for selecting the most similar user to the target user in order to obtain the target user profile. For instance, setting the threshold equal to 0.7, the nearest neighbours of u_j are selected as all users who have a similarity score larger than 0.7 in the *j-th* row in the user-to-user similarity matrix.

3.2 Predicted Scores

Recommendations are based on the probability that a user has a preference for a particular document. If this probability is high it is more likely that a recommendation on a document will be useful to the user. For the sake of completeness, we outline the notation used in this section.

Let u_j be the target user. Let $\boldsymbol{C}_{u_j}$ be the cluster of users in the nearest neighbours of u_j. Let i_q be the target document. Let $\boldsymbol{C}_{i_q,u_j}$ be the cluster of users in the nearest neighbourhood of u_j who downloaded i_q.

The predicted score that u_j has preferences for i_q is proportional to the probability that users in $\boldsymbol{C}_{u_j}$ have shown preferences for i_q.

$$P_{score}\left(i_q,u_j\right)=\frac{\left|\boldsymbol{C}_{i_q,u_j}\right|}{\left|\boldsymbol{C}_{u_j}\right|}, \tag{3}$$

where |.| is the cardinality of a set of users.

The predicted score is calculated based on the target user neighbourhood preferences on the target document.

4 Deployment

Our approach is implemented into a digital library experimental framework. The architecture of the integrated system is shown in Fig. 2. The digital library is supported on a data base that contains information on user registrations and

documents. The recommender system is supported on a data mart. Information such as document metadata, documents downloaded by user, and characteristic vectors is stored in the data mart. Recommendations are generated based on historical downloaded user's data. This is an unobtrusive approach and takes a downloaded document action as an implicit feedback.

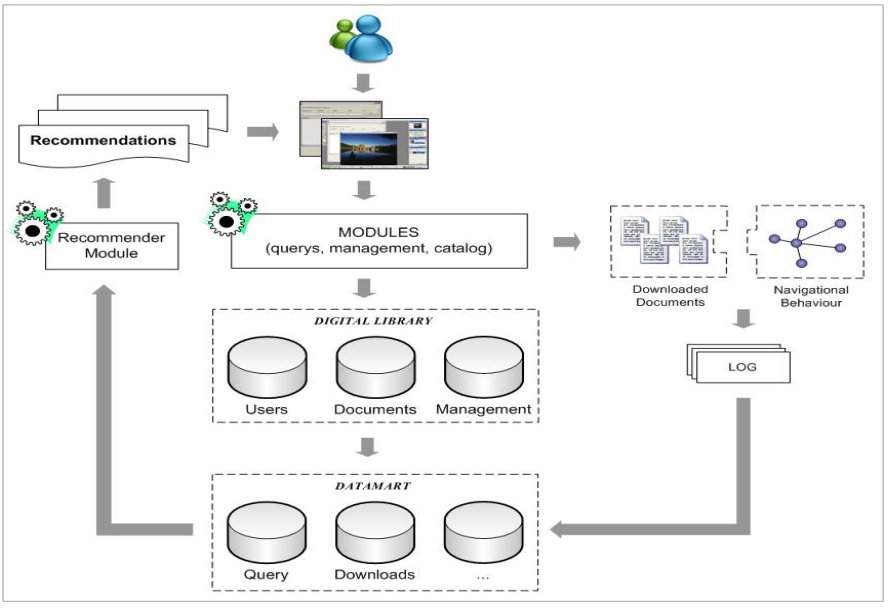


Fig. 2. System Architecture

Recommendations are twofold: a list of recommendations based on collaborative filtering and a second list based on downloaded document frequencies. Documents in the former, as in other recommender systems, are ranked according to their predicted score – in Equation 3. A verification process of previous recommendations is carried out to avoid making the same recommendations. However, collaborative filtering techniques provide recommendations regardless of users current interests. As for the second list, a user current interest is shown as his/her navigational behaviour. After his/her first search using key words, the main area of knowledge associated to those key words is used in a documents query and documents are sorted out by download frequency. A list of *n*-top documents is generated and a recommendation window is shown. Moreover, users have the choice of entering the window to check the recommendations included in the list or closing that window for the time being. Whatever courses of action a user takes are registered in the data mart.

5 Preliminary Evaluation

We have presented an approach to personalised information retrieval tasks in a digital library environment. According to Adomavicius *et al.* [1] the personalisation process

is integrated by three stages: understanding customers, delivering personalised offerings, measuring personalisation impact. In this paper, we focused on understanding users and delivering personalised offering phases. Moreover, once the digital library data mart contains enough information – in this specific domain – we will be able to evaluate our approach.

The performance of the asymmetric similarity measure is evaluated using the MovieLens data set, developed at the University of Minnesota [Available at http:// www.grouplens.org]. The dataset contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000. In a pre-processing phase, the most frequent rated genre, which is drama, was selected for the experimental validation.

We take as implicit rating the action of rating a movie. Information from u_1 to u_{100} was used in this evaluation. The asymmetric similarity measure was evaluated with the selected data set. The maximum similarity value reached between users was twelve times, as shown in Table 1. Users u_{53}, u_{70} and u_{77} rated the same movies that u_{21} rated. However, u_{53}, u_{70} and u_{77} rated more movies than u_{21} as the similarity score between u_{53} and u_{21} shows. In a similar manner, u_{33}, u_{58} and u_{88} rated the same movies that u_{94} did.

Table 1. MovieLens data set: List of users with the asymmetric similarity measure equal to one

Asymmetric Similarity Score	Asymmetric Similarity Score	Asymmetric Similarity Score	Asymmetric Similarity Score
$S(u_3,u_{62}) = 1$	$S(u_{62},u_3) = 0.3055560$	$S(u_{61},u_{48}) = 1$	$S(u_{48},u_{61}) = 0.0328947$
$S(u_7,u_{48}) = 1$	$S(u_{48},u_7) = 0.0526316$	$S(u_{61},u_{92}) = 1$	$S(u_{92},u_{61}) = 0.0515464$
$S(u_{20},u_{48}) = 1$	$S(u_{48},u_{20}) = 0.0394737$	$S(u_{94},u_{33}) = 1$	$S(u_{33},u_{94}) = 0.0122699$
$S(u_{21},u_{53}) = 1$	$S(u_{53},u_{21}) = 0.00340136$	$S(u_{94},u_{58}) = 1$	$S(u_{58},u_{94}) = 0.0150376$
$S(u_{21},u_{70}) = 1$	$S(u_{70},u_{21}) = 0.0588235$	$S(u_{94},u_{88}) = 1$	$S(u_{88},u_{94}) = 0.0416667$
$S(u_{21},u_{77}) = 1$	$S(u_{77},u_{21}) = 0.0588235$	$S(u_{98},u_{58}) = 1$	$S(u_{58},u_{98}) = 0.0601504$

Based on a user-to-user similarity matrix, the u_{100} neighbourhood was determined setting the threshold equal to 0.6; u_{100} rated 24 movies. The list of users belonging to u_{100} neighbourhood is shown in Table 2.

Table 2. MovieLens data set: List of users belonging to the u_{100} neighbourhood using the asymmetric similarity measure and a threshold equal to 0.6

The u_{100} Neighbourhood	Rated Movies	Asymmetric Similarity Score
u_{58}	113	0.791667
u_{48}	152	0.791667
u_{53}	294	0.666667
u_{62}	242	0.625000
u_{36}	111	0.625000

The list of users belonging to the u_{100} neighbourhood using 5-NN, based on a user-to-user similarity matrix calculated with the Jaccard similarity coefficient [8] is shown in Table 3. However, the u_{100} neighbourhood shown in Table 3 changes when a Jaccard is used due to the fact that the measure is affected by the large quantities of movies rated by users in Table 2.

Table 3. MovieLens data set: List of users belonging to the u_{100} neighbourhood using 5-NN

The u_{100} Neighbourhood	Rated Movies	Jaccard Score
u_{93}	23	0.270270
u_{13}	22	0.243243
u_{47}	20	0.222222
u_{95}	40	0.185185
u_{7}	8	0.185185

A prediction accuracy metric, the relative frequency with which the system makes correct decisions about whether a movie is of interest to a user, was used in the preliminary evaluation. Available information was divided into 90% training set and 10% cross-validation set. That is, when a user has 10 ratings, 9 ratings are used for building the model and 1 rating is used for validating the model.

When a list with 8 recommendations was generated for each user using the proposed approach, 78% of the users had rated at least one of the recommended movies in his/her cross-validation set. When a list with 8 recommendations was generated for each user using 5-NN, 56% of the users had rated at least one of the recommended movies in his/her cross-validation set. When a list with 8 recommendations was generated for each user using 10-NN, 47% of the users had rated at least one of the recommended movies in his/her cross-validation set.

6 Final Remarks

We had detected that the difference between document downloaded quantities may underestimate the similarity between users. We have proposed the use of an asymmetric similarity measure for reducing the impact of comparing users on the basis of the number of downloaded documents (large versus small quantities). The measure is used to identify a neighbourhood whose traits are strongly similar to those of an active user's behaviour thus reducing the possibility of generating irrelevant recommendations.

This approach has two characteristics: users' neighbourhood is dependent of a similarity score value rather than of a predefined number and a user does not always belong to the neighbourhood of the users who belong to his/her own neighbourhood. For generating recommendations, a prediction score is calculated based on the target user neighbourhood preferences on the target document.

Our next step, with data generated by users of the digital library, is to build a lifetime model for evaluating recommendations and to use a Bayesian approach for taking into account information no longer used in recommendations calculation, as *a priori* knowledge.

Acknowledgments. This work was partially supported by Instituto Colombiano para el Desarrollo de la Ciencia y la Tecnología "Francisco José de Caldas" (Colciencias) under Grant No. 1406-14-17902.

References

1. Adomavicius, G., Tuzhilin, A.: Toward a Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transaction on Knowledge and Data Engineering 17(6), 734–749 (2005)
2. Anderberg, M.: Cluster Analysis for Applications. Academic Press, New York (1973)
3. Billsus, D., Pazzani, M.: Learning Collaborative Information Filters. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 46–54 (1998)
4. Breese, J., Heckerman, D., Kadie, C.: Empirical Analysis of predictive algorithms for collaborative filtering. In: Proceedings of 14th Annual Conference on Uncertainty in Artificial Intelligence, pp. 43–52. Morgan Kaufman, San Francisco (1998)
5. Burke, R.: Hybrid Recommender Systems: Survey and Experiments. User Modelling and User-Adapted Interaction 12(4), 331–370 (2002)
6. Chen, H., Lynch, K.: Automatic Construction of Networks of Concepts Characterizing Document Databases. Journal of IEEE Transactions on Systems, Man and Cybernetics 22(5), 885–902 (1992)
7. Chen, H., Martínez, J., Ng, T., Schatz, B.: A Concept Space Approach to Addressing the Vocabulary Problem in Scientific Information Retrieval. An Experiment on the Worm Community System. Journal of the American Society for Information Science 48(1), 17–31 (1998)
8. Chen, P.Y., Popovich, P.M.: Correlation: Parametric and Nonparametric Measures. Sage University Papers Series: quantitative application in the social science (2002)
9. Choi, S.H., Kang, S., Jeon, J.Y.: Personalized recommendation system based on product specification values. Expert Systems with Applications 31, 607–616 (2000)
10. Garden, M., Dudek, G.: Mixed Collaborative and Content-Based Filtering with User-Contributed Semantic Features. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence, AAAI Press, Menlo Park, CA (2006)
11. Godoy, D., Amadi, A.: User profiling in personal information agents: a survey. The Knowledge Engineering Review 00(0), 1–33 (2005)
12. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An Algorithm Framework for Performing Collaborative Filtering. In: Proceedings of ACM SIGIR 1999, Berkeley, California, pp. 230–237 (1999)
13. Jin, R., Chai, J., Si, L.: An Automatic Weighting Scheme for Collaborative Filtering. In: Proceedings of SIGIR 2004, Sheffield, UK, pp. 337–344 (2004)
14. Kamahara, J., Asawaka, T., Shimojo, S., Miyahara, H.: A Community-based Recommendation System to Reveal Unexpected Interest. In: MMM 2005. Proceedings of the 11th International Multimedia Modelling Conference, pp. 433–438. IEEE Computer Society, Los Alamitos (2005)
15. Karpis, G.: Evaluation of Item-Based Top-N Recommendation Algorithms. In: Proceedings of the Tenth International Conference of Information and Knowledge Management, Atlanta, USA, pp. 247–254 (2001)
16. Kim, H., Kim, J., Herlocker, J.: Feature-Based Prediction of Unknown Preferences for Nearest-Neighbour Collaborative Filtering. Data Mining, 2004. In: ICDM 2004, pp. 435–438 (2004)

17. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J.: GroupLens: applying collaborative filtering to Usenet news. Communications of the ACM 40(3), 77–87 (1997)
18. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1), 76–80 (2003)
19. McLaughlin, M., Herlocker, J.: A Collaborative Filtering Algorithm and Evaluation Metric that Accurately Model the User Experience. In: Proceedings of SIGIR 2004, Sheffield, UK, pp. 329–336 (2004)
20. Montaner, M., Rosa, J.L.: A taxonomy of recommender agents on the Internet. Artificial Intelligence Review 19, 285–330 (2003)
21. Papagelis, M., Plexousakis, D., Rousidis, I., Theoharopoulos, E.: Qualitative Analysis of User-based and Item-based Prediction Algorithms for Recommendation Systems. Journal of Engineering Applications of Artificial Intelligence 18, 781–789 (2005)
22. Pennock, D.M., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In: Proceedings of the 16th Conference on Uncertainty and Artificial Intelligence, Stanford California, pp. 473–480 (2000)
23. Rashid, A.M., Lam, H., Karypis, G., Riedl, J.: ClustKNN A highly Scalable Hybrid Model-& Memory-Based CF Algorithm. In: Proceedings of WEBKDD 2006, Philadelphia, Pennsylvania USA (2006)
24. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work, Chapel Hill, NC USA, pp. 175–186 (1994)
25. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommendation algorithms for E-commerce. In: EC 2000. Proceedings of the 2nd ACM Conference on Electronic Commerce. Minneapolis, Minnesota, United States, pp. 158–167 (2000)
26. Schwab, I., Kobsa, A., Koychev, I.: Learning User Interests through Positive Examples Using Content Analysis and Collaborative Filtering. Draft from Fraunhofer Institute for Applied Information Technology, Germany (2001), Available at http://citeseer.ist.psu.edu
27. Shardanand, U., Maes, P.: Social Information Filtering: Algorithms for Automating "Word of Mouth". In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Denver Colorado, USA, pp. 210–217 (1995)
28. Shoukri, M.M.: Measures of Interobserver Agreement. Chapman Hall/CRC (2004)
29. Sugiyama, K., Hatano, K., Yoshikawa, M.: Adaptive Web Search based on user profile constructed without any effort from users. In: Proceedings of WWW 2004, New York, USA, pp. 675–684 (2004)
30. Teknomo, K.: Similarity Measurement, http://people.revoledu.com/kardi/tutorial/Similarity/
31. Trujillo, M., Millan, M., Ortiz, E.: A Recommender System Based on Multi-features. In: Gervasi, O., Gavrilova, M.L. (eds.) ICCSA 2007. LNCS, vol. 4706, pp. 370–382. Springer, Heidelberg (2007)
32. Wang, J., de Vries, A., Reinders, M.: Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. In: SIGIR 2006. Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval Seattle, Washington, USA, pp. 501–508 (2006)
33. Xi, W., Fox, E., Fan, W., Zhang, B., Chen, Z., Yan, J., Zhuang, D.: SimFusion: Measuring Similarity using Unified Relationship Matrix. In: Proceedings of SIGIR 2005, Salvador, Brazil, pp. 130–137 (2005)
34. Xue, G., Lin, C., Yang, Q., Xi, W., Zeng, H., Yu, Y.: Scalable Collaborative Filtering Using Cluster-based Smoothing. In: Proceedings of SIGIR 2005, Salvador, Brazil, pp. 114–121 (2005)

Stop Wasting Time: On Predicting the Success or Failure of Learning for Industrial Applications

J.E. Smith and M.A. Tahir

School of Computer Science
University of the West of England
Bristol, BS161QY, UK
{james.smith,muhammad.tahir}@uwe.ac.uk

Abstract. The successful application of machine learning techniques to industrial problems places various demands on the collaborators. The system designers must possess appropriate analytical skills and technical expertise, and the management of the industrial or commercial partner must be sufficiently convinced of the potential benefits that they are prepared to invest in money and equipment. Vitally, the collaboration also requires a significant investment in time from the end-users in order to provide training data from which the system can (hopefully) learn. This poses a problem if the developed Machine Learning system is not sufficiently accurate, as the users and management may view their input as wasted effort, and lose faith with the process. In this paper we investigate techniques for making early predictions of the error rate achievable after further interactions. In particular we show how decomposing the error in different components can lead to useful predictors of achievable accuracy, but that this is dependent on the choice of an appropriate sampling methodology.

1 Introduction

The successful application of machine learning techniques to industrial problems places various demands on the collaborators. Vitally, the collaboration also requires a significant investment in time and commitment from the end-users in order to provide training data from which the system can (hopefully) learn. This poses a problem if the developed Machine Learning system is not sufficiently accurate, as the users and management may view their input as wasted effort, and lose faith with the process.

A significant factor that would help in gaining confidence and trust from end-users would be the ability to quickly and accurately predict whether the learning process was going to be successful. Perhaps more importantly from a commercial viewpoint, it would be extremely valuable to have an early warning that the user can save their effort while the system designer refines the choice of data, algorithms etc. In a system applied in industrial application, it is necessary that the learning phase is as short as possible and it is essential that the system can tell by itself and as early as possible whether the learning process will be

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 673–683, 2007.

successful. In some cases such as random training input by the operators or missing information the system will not be able to successfully complete the learning process.

In this paper we investigate techniques for making early predictions of the error rate achievable after further interactions. We will consider that we are given N samples, and that the system is still learning and refining its model at this stage. We are interested in predicting what final accuracy might be achievable if the users were to invest the time to create M more samples. In particular we focus on the following aspects:

- What are suitable descriptors of the system's behaviour after some limited number N of samples?
- What is the most appropriate measure of the system's predictive accuracy after further training with $N + M$ samples?
- Is it possible to find useful relationships for predicting the second of these quantities from the first?
- What is the effect of different choices of methodology or sampling regime to estimate these quantities?

In general the error will be a complicated function, but the hypothesis of this paper is that we can deal with it more easily if we decompose it into a number of more stable functions. Therefore this paper concentrates on the use of the well-known bias-variance decomposition as a source of predictors [1,2]. Although we will use results from a wide range of classifiers, for the purposes of this paper we will take them one at a time, rather than considering heterogenous ensembles of classifiers with different biases. We will also take the pragmatic approach of constructing an "early warning system". In other words, rather than trying to predict the absolute value of the final accuracy, we will consider ways of estimating upper bounds on the accuracy achievable.

The rest of this paper proceeds as follows. In Section 2 we review related work in the field, in particular the bias-variance decomposition of error that we will use. Following that in Section 3 we discuss various suggested methods for using the available data to estimate the various quantities involved. Section 4 describes our experimental methodology and Section 5 describes and discusses the results obtained. Finally in Section 6 we draw some conclusions and suggestions for further work.

2 Bias-Variance Decomposition: A Review

A number of recent studies have shown that the decomposition of a classifier's error into bias and variance terms can provide considerable insight into the prediction of the performance of the classifier [1,2]. Originally, it was proposed for regression [3] but later, this decomposition has been successfully adapted for classification [1,2,4]. While a single definition of bias and variance is adopted for regression, there is considerable debate about how the definition can be extended to classification [1,5,6,7,8,9]. In this paper, we use Kohavi and Wolpert's [1]

definition of bias and variance on the basis that it is the most widely used definition [10,11], and has strictly non-negative variance terms.

2.1 Basic Definitions of Bias, Variance and Noise

Kohavi and Wolpert define bias, variance and noise as follows [1]:

Squared Bias: "This quantity measures how closely the learning algorithm's average guess (over all possible training sets of the given training set size) matches the target".

Variance: "This quantity measures how much the learning algorithm's guess bounces around for the different training sets of the given size".

Intrinsic noise: "This quantity is a lower bound on the expected cost of any learning algorithm. It is the expected cost of the Bayes-optimal classifier".

2.2 Kohavi and Wolpert's Definition of Bias and Variance

For a particular target function f and a size of the training set m, the expected misclassification rate E(C)(an error has cost 1 and a correct prediction cost 0) is defined as

$$E(C) = \sum_x P(x)(\sigma_x^2 + bias_x^2 + variance_x) \tag{1}$$

where

$$bias_x^2 = \frac{1}{2}(1 - \sum_{y \epsilon Y}[P(Y_F = y|x) - P(Y_H = y|x)]^2)$$

$$variance_x = \frac{1}{2}(1 - \sum_{y \epsilon Y} P(Y_H = y|x)^2)$$

$$\sigma_x^2 = \frac{1}{2}(1 - \sum_{y \epsilon Y} P(Y_F = y|x)^2)$$

Here x ranges over the instance space X, Y is the predicted variable with elements $y \epsilon \{0, 1\}$ [12]. The actual target function f is a conditional probability distribution and the hypothesis or model h generated by learner is also conditional probability distribution $P(Y_H = y|x)$. Although not clear from the equations, the conditional events in the conditional probabilities are parameterised over f and m. In other words, $P(Y_H = y|x)$ must be rewritten as

$$P(Y_H = y|f, m, x) = \sum_d P(d|f, m) P(Y_H = y|d, x) \tag{2}$$

where $P(d|f, m)$ is the probability of generating training sets d from the target f, and $P(Y_H = y|d, x)$ is the probability that the learning algorithm predicts y for point x in response to training set d.

2.3 Bias as an Upper Limit on Accuracy

An alternative perspective on the above analysis is that the bias term reflects an inherent limit on a classifier's accuracy resulting from the way in which it forms decisions boundaries. For example even in a two-dimensional space, an elliptical class boundary can never be exactly replicated by a classifier which divides the space using axis-parallel decisions. Therefore we can treat the sum of the inherent noise and bias terms as an upper limit on the achievable accuracy for a given classifier. A number of studies have been made confirming the intuitive idea that the size of variance term drops as the number of training samples increases, whereas the estimated bias remains more stable, e.g. [2]. Please note that in many prior works it is assumed that the inherent noise term is zero, and also for a single classifier it is not possible to distinguish between inherent noise and bias, so we will adopt the convention of referring to these collectively as bias.

The hypothesis of this paper is that if we can estimate the value of the bias term it will form an accurate predictor to bound the error rate observed after more training examples. The way that we will do this for a given size sample N is to repeatedly draw test and training sets from the sample and observe what proportion of the items are always misclassified, what proportion are sometimes misclassified, and what proportion are never misclassified. As we will next discuss, this raises the issues of how we should do this repeated process.

3 Prediction Methodology

As discussed in Section 2, a number of recent studies have shown that the decomposition of a classifier's error into bias and variance terms can provide considerable insight into the prediction of the performance of the classifiers [1,2]. However, identifying the quantities that we wish to measure merely leads us to the next question - what is the most appropriate methodology for estimating the values of those quantities?

To give a simple example of why this is important, the hypothesis of this paper relies on being able to distinguish between those data items that are **always** going to be misclassified by a given classifier, and those which will **sometimes** be misclassified, depending on the choice of training set. Since the well known $N-$fold cross-validation approach only classifies each data item once, it does not permit this type of decomposition and cannot be used. Luckily alternative approaches have been identified and studies by other authors. Leveraging this work, in this paper, we will compare the approaches proposed by Kohavi & Wolpert [1] and Webb & Conilione [10].

Kohavi & Wolpert Hold-out Procedure: Kohavi & Wolpert used a holdout procedure for estimating the bias and variance of a classifier C from a dataset D. In their approach, samples D are randomly divided into 2 parts: Training samples T_r and Testing samples T_e. T_r samples are further divided into N training sets $t_{r_1}, t_{r_2}, ..., t_{r_n}$ by uniform random sampling without replacement. To get training set of size m, they chose T_r to be size $2m$. That allows $(\frac{2m}{m})$ different

possible training sets; and thus guarantees that there are not many duplicate training sets in the set of N training sets; even for small values of m. Each classifier is trained using each training sets and bias and variance are estimated using test set T_e. The outcome of this is a set of N class precisions for each of the elements in the test set.

Webb & Conilione sub-samples Cross Validation Procedure: In the second set of experiments, we will decompose error into bias-variance using sub-sampled cross-validation proposed by Webb & Conilione [10] but using same definitions for bias and variance as above. Webb & Conilione have argued that hold out approach proposed in [1] is fundamentally flawed and resulting in small training sets and thus provide instability in the estimates it derives. They proposed that sub-sampled cross-validation (CV) procedure is superior to both the holdout and bootstrap procedures and thus provides greater degree of variability between training sets. Webb's procedure repeats N-Fold CV l times. This ensures that each sample x of the dataset D is classified l times. The $bias_x$ and $variance_x$ can be estimated from the resulting set of classifications. The final bias and variance is estimated from the average of all $x \epsilon D$ [10,11].

4 Experimental Methodology

Choice of Classifiers: Ten different classification algorithms are selected each with different bias and variance characteristics namely: Naive Bayes [13], Decision Tree [15], Nearest Neighbor [16], Bagging [18], AdaBoost [19], Random Forest [20], Decision Table [21], Bayes Network [13], Support Vector Learning [22], and Ripple-Down Rule learner [17]. All these classifiers are implemented in WEKA library [17].

Data sets: The experiments are carried out on the following Four Artificial and Five Real-World Surface Inspection data sets described in Table 1. Each artificial dataset consists of 13000 contrast images with a resolution of $128 * 128$ pixels. The good/bad labels were assigned to the images by using different sets of rules of increasing complexity. The proposed prediction analysis is also evaluated out on real world data sets of CD-print and Egg inspection. The data set for CD print consists of 1534 images and each image is labeled by 4 different operators. Thus, 4 different CD print data sets are available. From each set of images, we derive 2 feature vectors (FVs) consisting of 17 and 74 features respectively. The first FV contains only image-level information while second FV also contains features from objects within the image.

Trend Line using Linear Regression: Linear regression is a statistical tool used to predict future values from past values. Regression lines can be used as a way of visually depicting the relationship between the independent (x) and dependent (y) variables in the graph. A straight line depicts a linear trend in the data. In this paper, we will use linear trend line between bias (For first N samples only) and error (For $N + M$ samples) to predict the Success or Failure

Table 1. Datasets Description

Name	Samples	Description
Artificial 1-3	13000	Used for Linear Regression Analysis
Artificial 4	13000	Used after Linear Regression Analysis for Prediction
CD Print Op1-Op3	1534	Used for Linear Regression Analysis
CD Print Op4	1534	Used after Linear Regression Analysis for Prediction
Egg	4238	Used for Linear Regression Analysis

of Learning for Industrial Applications. We will use squared Pearson correlation coefficient R^2 as a measure to analyze the quality of prediction. The closer R^2 is to 1.0; the better is the prediction. This is of course an extremely simple way of measuring the relationship between estimated bias and final error, and more sophisticated techniques exist in the fields of statistics and also Machine Learning. However, as the results will show it is sufficient for our purposes. An obvious candidate for future work is to consider approaches which will give us confidence intervals on the predicted error for a given observed bias, as this will fit in better with the concept of providing an upper bound on the achievable accuracy.

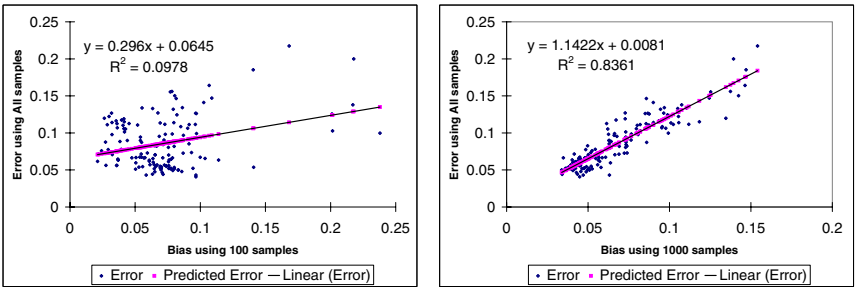
5 Results and Discussion

As discussed in Section 1, we have estimated the bias using $\{100, 200, 300, ...1000\}$ samples and then the error using all samples of artificial/real data sets by both Kohavi and Webb sampling approaches.

Results with Kohavi's sampling procedure: Figure 1 shows linear regression analysis for bias-error when Kohavi's approach is used for bias-error decomposition. Bias is estimated using 100 and 1000 samples respectively. 7 data sets are used for regression analysis (3 Artificial data sets, CD Print labeled by 3 operators, and 1 Egg data set). Each data set consists of 2 different feature vectors and is evaluated using 10 classes as discussed in Section 4. The goodness of fit of regression model is measured using Correlation R^2. As clearly indicated from these graphs, R^2 is very low when model is fit using only 100 samples while correlation is high when 1000 samples are used. Furthermore, straight line using 1000 samples depicts a linear trend in the data.

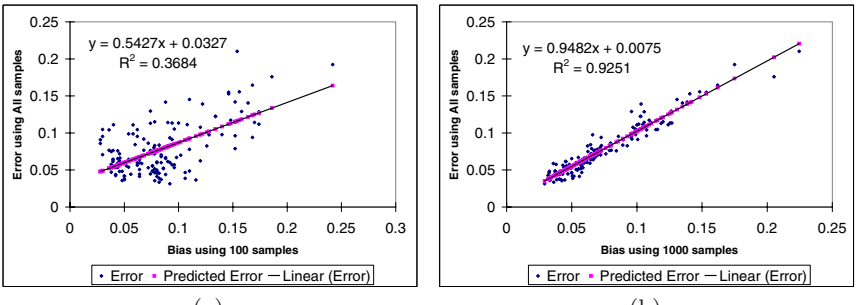
Results with Webb's sampling procedure: Figure 2 shows linear regression analysis for Bias-Error when Webb's approach is used for bias-error decomposition. Again, as clearly indicated from these graphs, the R^2 is very low when model is fit using only 100 samples while correlation is high when 1000 samples are used. However, these values are consistently higher than those obtained using Kohavi's approach.

Stability of Predictions: Figure 3 shows the graph indicating relationship between varying number of samples {100,200,300,...,1000} and {R^2, x-coefficient, intercept} for both Kohavi's and Webb's approaches. This shows how rapidly



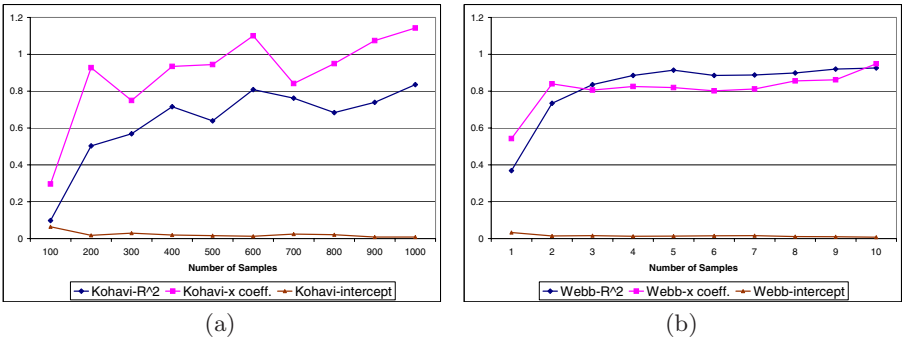
(a) (b)

Fig. 1. Graphs showing linear regression analysis for Bias-Error using 100 and 1000 samples respectively. Kohavi approach is used for bias-error decomposition.



(a) (b)

Fig. 2. Graphs showing linear regression analysis for Bias-Error using 100 and 1000 samples respectively. Webb approach is used for bias-error decomposition.



(a) (b)

Fig. 3. Number of samples vs $\{R^2$,x-coeff.,intercept$\}$ using (a) Kohavi's (b) Webb's bias-error decomposition

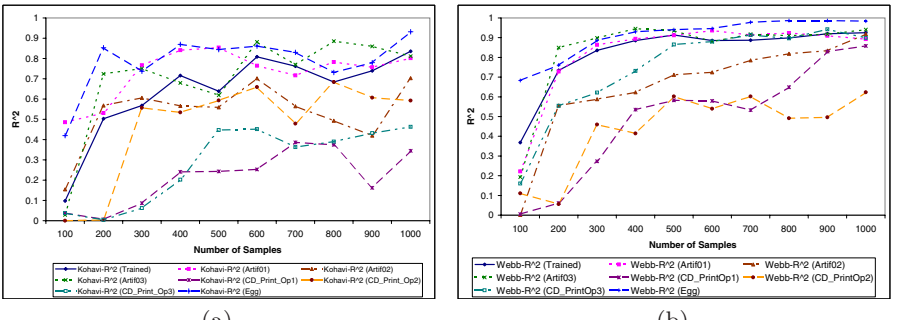
the linear regression equation stabilizes in these two cases. It is clear from the graph that correlation using Webb approach is high and more stable. One of the

reasons that Kohavi's approach is not stable is the use of hold out approach. It has been argued that in Kohavi's approach, samples are randomly divided into training and testing pools and then training pool is further divided into training sets and that can results in instability in the estimates [11]. Another explanation is that a single test set is chosen from the available samples. For small sample sizes this may not always contain sufficiently representative set of items so successive test sets might be "easy" or "hard".

Prediction Testing using Trained/Unseen Datasets from Trained Regression Model: In our experiments, R^2 is used as a measure to evaluate the goodness of regression models. Another way to evaluate the goodness of regression models is as follows:

- For each combination of the ten classifiers, the seven data sets used in training, the two unseen data sets, and the ten sample sizes we repeat the following:
  - estimate the bias component of the error using both Kohavi and Webb's approaches
  - plug this value into the regression equations obtained above to get a predicted final error
  - compare this to the observed final error
- these values can now be subjected to a new regression analysis to see how well the predictions correlate to observed error from the full dataset.

Figures 4 and 5 show the relationship between number of samples and R^2. Again, it is clear from both these figures that R^2 using Webb approach is high and stable. Artif04 has approximately the same R^2 as that of trained one especially when samples are greater than 500 by regression equation using Webb sampling approach. Correlation is quite low for CD-Print-Op2 and CD-Print-Op4 data sets. Closer inspection of the results shows that in almost every case the observed error from 1534 samples of the CD print data is higher than that predicted by inserting the bias observed from fewer samples into the regression



(a) (b)

Fig. 4. Number of samples vs R^2 for trained data using (a) Kohavi's (b) Webb's approach

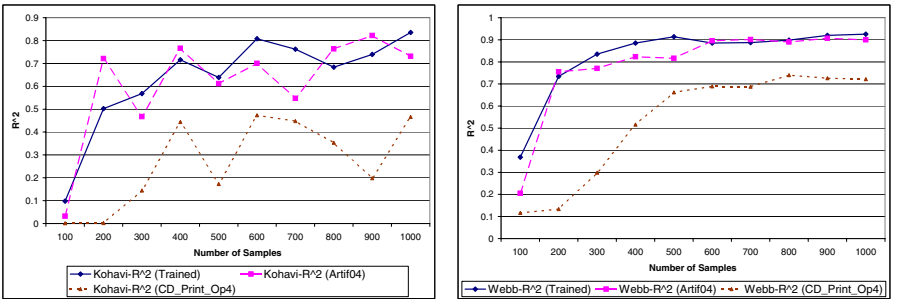


Fig. 5. Number of samples vs R^2 for unseen data using (a) Kohavi's (b) Webb's approach

equation. This is because the observed error still contains a significant component of variability due to the effects of the relatively small training and test sample sizes. By contrast, for the artificial data sets, where we have nearly ten times more data, the variance components have almost disappeared and so our predictions correlate highly to the observed error. This illustrates our earlier point - that the predictions we are making here have to be treated as upper bounds on the achievable accuracy.

6 Conclusion

In this paper, we have investigated techniques for making early predictions of the error rate achievable after further interactions. Linear trend line between bias and error is used to predict the Success or Failure of Learning for Industrial Applications. The experiments are carried out on the Artificial and Real-World data sets. We have shown that squared Pearson correlation coefficient R^2 is a good measure to analyze the quality of prediction. We have also shown that Webb's approach allows much more accurate and stable estimates of error components. These results are valid for ten very different forms of classifier used in this paper. As the high correlation between the long-term observed error, and the predictions for that based on observed bias after 1000 samples shows, the (bias+implicit noise) term of the error stabilises rather quickly for all of the different types of classifier tested. Thus it forms not only a good upper bound on the achievable accuracy, but also a good estimator for the final accuracy provided enough samples are available for the variance term to decrease.

This is of course an extremely simple way of measuring the relationship between estimated bias and final error, and more sophisticated techniques exist in the fields of statistics and also Machine Learning. However, as the results will show it is sufficient for our purposes. An obvious candidate for future work is to consider approaches which will give us confidence intervals on the predicted error for a given observed bias, as this will fit in better with the concept of

providing an upper bound on the achievable accuracy, that can be used as an "early warning" of impeding failure, so that users' confidence can be maintained.

Acknowledgements

This work was supported by the European Commission (project Contract No. STRP016429, acronym DynaVis). This publication reflects only the authors' views.

References

1. Kohavi, R., Wolpert, D.H.: Bias Plus Variance Decomposition for Zero-One Loss Functions. In: Proceedings of the 13th International Conference on Machine Learning (1996)
2. Brian, D., Webb, G.I.: On the effect of data set size on bias and variance in classification learning. In: Proceedings of the 4th Australian Knowledge Acquisition Workshop, pp. 117–128 (1999)
3. Geman, S., Bienenstock, E., Doursat, R.: Neural Networks and the bias/variance dilemma. Neural Computation 4, 1–48 (1995)
4. Rodriguez, J.J., Alonso, C.J., Prieto, O.J.: Bias and Variance of Rotation-based Ensembles. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 779–786. Springer, Heidelberg (2005)
5. Breiman, L.: Bias, variance, and arcing classifiers, Technical report 460, Statistics Department, University of California, Berkeley, CA
6. Domingos, P.: A unified bias-variance decomposition and its application. In: Proceedings of the 17th International Conference on Machine Learning, Stanford University, USA, pp. 231–238 (2000)
7. Friedman, J.H.: On bias, variance, 0/1-loss, and the curse of dimensionality. Data Mining and Knowledge Discovery 1(1), 55–77 (2000)
8. James, G.: Variance and bias for general loss functions. Machine Learning 51(2), 115–135 (2003)
9. Kong, B.E., Dietterich, T.G.: Error-correcting output coding corrects bias and variance. In: Proceedings of the 12th International Conference on Machine Learning, pp. 313–321. Morgan Kaufmann, San Francisco (1995)
10. Webb, G.I.: Multiboosting: A technique for combining boosting and wagging. Machine Learning 40(2), 159–196 (2000)
11. Webb, G.I., Conilione, P.: Estimating bias and variance from data (2003) (Under Review), http://www.csse.monash.edu.au/~webb/Files/WebbConilione03.pdf
12. Putten, P.I.D., Someren, M.V.: A Bias-Variance Analysis of a Real World Learning Problem: The CoIL Challenge 2000. Machine Learning 57, 177–195 (2004)
13. Duda, R O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley Interscience, New York (2000)
14. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc, San Francisco (1993)
15. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(1), 4–37 (2000)

16. Cover, T.M., Hart, P.E.: Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory 13(1), 21–27 (1967)
17. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
18. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
19. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of International Conference on Machine Learning, pp. 148–156 (1996)
20. Breiman, L.: Random Forests. Machine Learning 45(1), 5–32 (2001)
21. Kohavi, R.: The Power of Decision Tables. In: Proceedings of the 8th European Conference on Machine Learning (1995)
22. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: Schoelkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, MIT Press, Cambridge (1998)

Parallel Wavelet Transform for Spatio-temporal Outlier Detection in Large Meteorological Data

Sajib Barua[1] and Reda Alhajj[1,2]

[1] Computer Science Dept, University of Calgary, Calgary, Alberta, Canada
[2] Department of Computer Science, Global University, Beirut, Lebanon
{baruas,alhajj}@cpsc.ucalgary.ca

Abstract. This paper describes a state-of-the-art parallel data mining solution that employs wavelet analysis for scalable outlier detection in large complex spatio-temporal data. The algorithm has been implemented on multiprocessor architecture and evaluated on real-world meteorological data. Our solution on high-performance architecture can process massive and complex spatial data at reasonable time and yields improved prediction.

1 Introduction

This paper introduced a novel approach to locate outlier in meteorological data, which are collected over time and space. Meteorological data are spatio-temporal data because of their multidimensional properties, structures (geometrical shape), distribution over space and variation with time. Techniques for accommodating semantic (identity and attributes), spatial (geometry, location and topology), and temporal (time of occurrences and observation) properties of meteorological events try to resolve theoretical and computational issues concerning the spatial modeling of meteorological data. Processing meteorological data to forecast severe events has always been a challenging problem over the last decades because of several reasons: 1) the variability found in the properties of the meteorological data; 2) the data is complex, massive, and has spatial properties; 3) the solution strategy varies with the location and type of the collected data. Data mining has demonstrated enormous potential as a processing tool for meteorological data. While major data mining techniques try to find general patterns in the data set, outlier or deviation detection draws attention to finding exceptional, dissimilar and inconsistent patterns compared with the rest of the data. An object is defined as a spatio-temporal outlier if its non-spatial attribute is significantly different from its neighbors in spatial relation and/or from other observations in a given time period. As weather data is concerned, outliers indicate anomalies or severe events such as tornadoes and forest fires that tend to occur in an area. Therefore, identifying spatio-temporal outliers is an essential problem with both scientific and social impacts. Neglecting outliers may lead to disasters like those that lastly happened in Asia and the United states.

Mining meteorological data for identifying spatio-temporal outliers needs to consider the following aspects in general: 1) high dimensionality with arbitrary

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 684–694, 2007.

and complex data types and large data sets at high resolution; 2) the data is geographically distributed over space and time.

Although many works are done in spatial outlier detection, e.g., [5,9,10,12], very few of them focus on spatio-temporal outliers. For instance, Tao *et al* [4] extended the spatial outlier mining to capture the semantic and dynamic aspects of spatio-temporal data in multi scales. They adopted multi-resolution clustering algorithm based on semantic knowledge of ST-objects and applied them to find outliers in multi-scale property of geographic phenomena. Birant *et al* [3] improved DBSCAN algorithm to capture the temporal aspects of ST-objects and introduced a scale to measure density of each cluster. Ramachandran *et al* [8] developed a flexible framework ADaM.

High performance computational architecture not only allows analysis at multiple spatial scales and locales, but also improves more realistic simulation of extreme events at high resolution to provide climate information for resource management and impact assessment. In this direction, wavelet transform (WT) can locate the frequency change (property variation) because of its multi-resolution and multi-scale properties; hence performs well in detecting the boundaries of outlier regions. Weather data (temperature, vapor distribution, rainfall) has spatial properties because of its multidimensional features and geometric pattern. Therefore, we adopt Parallel Wavelet Transform (PWT) algorithm for outlier detection in large scale spatial data to take the advantage of the computational power on the multiprocessor machine. Our parallel approach also tries to tolerate the architectural latencies by overlapping communication with computation.

The rest of the paper is organized as follows. Section 2 presents a framework for outlier mining using parallel wavelet transform. Section 3 presents experimental results. Section 4 is conclusions.

2 Parallel Spatio-temporal Outlier Detection

In the real atmosphere, anomalies occur at different spatial and temporal scales. Therefore, our task involves defining a region outlier as a group of adjoining points whose features are inconsistent with those of their surrounding neighbors or time frame. Spatial properties can not be analyzed with a uniform scale. Wavelet analysis allows the analysis data at different scales for two important properties. 1) **Multi-resolution**: wavelet analysis uses stretched wavelet to capture the coarser view and compressed wavelet to capture the detailed information in signal; 2) **Localization of the frequency**: WT can also detect the frequency change in the time domain.

Spatial outliers appear as small regions; hence are local outliers in most cases. On the other hand, temporal outliers are defined based on the duration of the time domain. Detection of events like El Niño and La Niña focus on outliers in long time period, whereas climatical changes in a month focus on finding local outliers. We use Daubechies $D4$ WT to analyze the data.

2.1 Calculation of Daubechies $D4$ Wavelet Transform

The $D4$ transform has four wavelet and scaling function coefficients: $h_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}$,$h_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}$, $h_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}$, and $h_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}$. The wavelet function coefficient values are: $g_0 = h_3$, $g_1 = -h_2$, $g_2 = h_1$, and $g_3 = -h_0$. For a data set of N values, the scaling function calculates $\frac{N}{2}$ smoothed values, and the wavelet function calculates $\frac{N}{2}$ differences at each step of the wavelet transform. In the ordered wavelet transform, the smoothed values are stored in the lower half of the N elements input vector. Each step of the wavelet transform applies the wavelet function to the input data. If the original data set has N values, the wavelet function will be applied to calculate $\frac{N}{2}$ differences (reflecting change in the data). In the ordered wavelet transform, the wavelet values are stored in the upper half of the N elements input vector. The scaling and wavelet functions are calculated by taking the inner product of the coefficients and four data values. Daubechies $D4$ scaling function is $s'_i = h_0 s_i + h_1 s_{i+1} + h_2 s_{i+2} + h_3 s_{i+3}$ and Daubechies $D4$ wavelet function is $s'_{i+1} = g_0 s_i + g_1 s_{i+1} + g_2 s_{i+2} + g_3 s_{i+3}$.

Forward Transform: Each iteration in WT calculates a scaling function value and a wavelet function value. The index i is incremented by two with each iteration, and new scaling and wavelet function values are calculated. In case of the forward transform, with a finite data set, i is incremented until it is equal to $N-2$. In the last iteration, the inner product is calculated from $s[N-2]$, $s[N-1]$, $s[N]$ and $s[N+1]$. Since $s[N]$ and $s[N+1]$ don't exist (they are beyond the end of the array), this presents a problem.

2.2 Parallel Daubechies Fast Wavelet Transform

Different authors worked on the design and implementation of classical wavelet construction. Our implementation is on distributed memory architecture, where each processor has fast access to its own memory. The important issues for an efficient parallel algorithm are load balancing and communication latency, which refer to an efficient distribution of task and data over multiple processors to minimize the intercommunication among them.

Data Distribution: The data distribution strategy for a parallel algorithm is greatly affected by the computation pattern at each stage. In sequential algorithm, we observe that at each level, input data are generated from the low pass filter of the previous level. After each level computation, redistribution on the data is done to bring the results from the low pass filter consecutive and they appear as input for the next level. Redistribution ensures data locality for next level computation and reduces inter-communication cost if data required for computation resides in remote processors. But, it requires data swapping among processors, which incurs a huge amount of communication and load balancing will be poor. The number of input data is halved at each level; hence, such kind of redistribution will only bring the input data local to some of the processors, leaving the rest of them totally idle.

We avoid the redistribution and achieve a proper load balancing by assigning equal number of inputs to each processor. If there are N number of inputs and P number of processors, each processors in blocked data distribution will get $\frac{N}{P}$

number of inputs. At each level, the number of inputs is reduced by half and the process is continued until one input is available. Assuming N and P as powers of 2, the total number of levels required is $\lg N$.

Computation: The computational cost to get a new value is 4 multiplications and 1 summation. On single processor, for N values each stage requires $4N$ multiplications and N summations. Whereas if we employ P processors, total number of multiplications and summations will be $4\frac{N}{P}$ and $\frac{N}{P}$, respectively, which is a great reduction in computational cost. N should be greater than P and for better performance, at initial stage we assume, each processor should have at least 4 input values. As we continue, the number of input values for the next stage will be reduced by half. The total number of stages which have at least one input value are $\lg \frac{N}{P} + 1$. Among them, at the last stage each processor gets only one input value and the rest get more than one values. For $(\lg N - (\lg \frac{N}{P} + 1)$ stages, not all processors have values for computation and will remain idle. Therefore, we have load balancing in major number of stages.

Communication: Each processor needs one send and one receive. But, when each processor has only one element to compute, the number of sending or receiving will be 3 as sent or received values are directed to or coming from 3 different processors. Therefore, $(\lg N - \lg \frac{N}{P}) = \lg P$ number of stages require 3 sending and 3 receiving if the processor has value to compute. As this number increases with the increase of P, therefore for a data set (N), a specific value of P will give better load distribution, where $\lg \frac{N}{P}$ will be higher and communication cost (for $\lg P$ stages) will never be dominating over the computational cost.

2.3 Parallel Outlier Mining

The wavelet analysis algorithm uses a parallel wavelet algorithm on climate data in order to discover regions with prominent spatial or temporal variation at different scales. A set of scales is provided by the domain experts beforehand. We are also given the beginning and ending latitude for our analysis. The wavelet analysis is performed on the dataset recorded along the latitude range to discover regions with prominent spatial variation at different scales. The wavelet value for each data point is compared against a threshold value (w). If the value is higher than w, we consider the data at that point as suspected outlier and record the location (latitude, longitude). In spatial domain, suspected outliers are grouped together using Z-value approach [11] and form outlier region.

```
WAVELET ANALYSIS FOR OUTLIER MINING
Input:-- t1: the beginning latitude (or time); tn: the ending latitude (or time);
         S: a set of selected scales; w: a pre-defined threshold wavelet power;
         X: a given data set; SuspectSet: a set of points to be outliers;
Output:-- Y: a set of outlier regions;
/*Calculation of wavelet power along all latitudes or time*/
for ( i = t1; i < tn ; i++){
   wTransform = ParallelWTransform(X,S,i);}
/* Identify point which wavelet power is higher than the thresold*/
for each p in wTransform{
   if (p > w){CandidateOutlierSet(p, CandidateSet)}}
/* Form regions using statistical approach*/
Y = FormRegionOutlier(CandidateSet); Output(Y);
```

3 Experimental Result

We conducted our experiment on sea surface temperature (SST) dataset collected from National Climatic Data Center (NCDC) NOMADS LAS server. We performed the spatial analysis of daily optimum interpolation (OI) SST on a $\frac{1}{4}$ degree grid at different locations (e.g., equatorial pacific). The temporal analysis is performed on real-time SST data from moored ocean buoys collected from Tropical Atmosphere Ocean (TAO) project. Our temporal outlier detection tested SST data at different temporal resolutions from 10 minutes to daily basis for locations on the equator. The tests have been conducted on WestGrid. The PWT algorithm was implemented in C with MPI library and tested on a distributed platform nexus et el. hosted by the University of Alberta. This is a collection of SGI SMP machines, ranging up to 256 processors; for our experiment, we used 32 processors. Finally, we present the data analysis and then show the scalability of the outlier detection algorithm on parallel architecture for data at high resolution spatio-temporal scale.

3.1 Spatial Analysis

In meteorological data, outliers appear as arbitrary small shapes or regions (eye of a cyclone), where a sharp temperature or vapor distribution change occurs. To compute this climatological change for a location, we need to identify its neighborhood and compute the feature difference. In GIS, spatial resolution is defined as the ability to define sharply and clearly the extent or shape of features within an image or area on the ground by an imaging system, such as a satellite sensor. At the highest resolution, more details become clear, and the information is sharper. Spatial scale is used to define the resolution. Such division is more or less arbitrary; for example, micro is the smallest unit which may involve 1° degree or $1m - 1km$ scope; and mega is assigned the global scope, which may involve the whole planet. We worked on data gathered at $\frac{1}{4}^\circ$ interval across latitude, and each point on the latitude is the average SST of the 2° north and south longitudinal positions. As our focus is on latitude climatology change, for a location the neighborhood encompasses 1° east and west across latitude, and 2° north and south across longitude. For a latitude-longitude plan, we apply our algorithm for each latitude at 2° intervals and combine the outlier regions.

Fig. 1 plots the SST temperature over South Pacific (Fig. 2) recorded on Nov 28, 2006; we have shown some of the outlier locations (A, B, C and D) in rectangles. Each of A, B and C is local outlier compared to its neighborhood region, and C and D can be considered as global outliers for their highest and lowest value, respectively. These locations are more obvious from the contour plot (Fig. 3) collected from NOAA/PMEL. Our algorithm can find these outlier locations from the wavelet decomposition in time-scale map. Sharp changes (high frequency or bursts) in the temperature are captured at low scales and higher time resolution window. Whereas a trend (low frequency), such as a particular temperature continuing for a longer period, is captured at high scale and poor time resolution window. In Fig. 4, we observe such phenomena in approximation

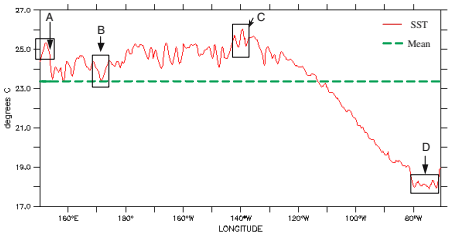


Fig. 1. Daily optimum interpolation (OI) SST ($^\circ C$): time 01-Nov-06

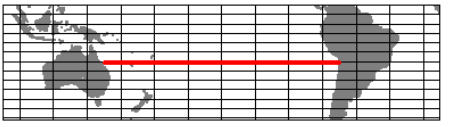


Fig. 2. South pacific: lat $20.6^\circ S$, long: $150.1^\circ E$ to $70.4^\circ W$

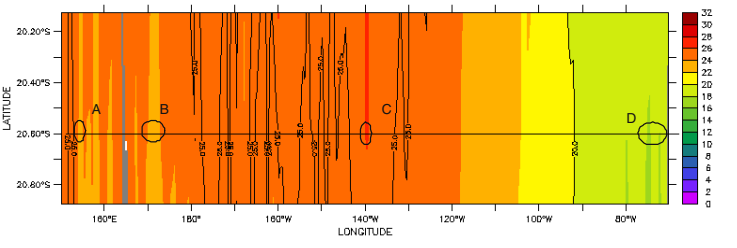


Fig. 3. SST contour plot

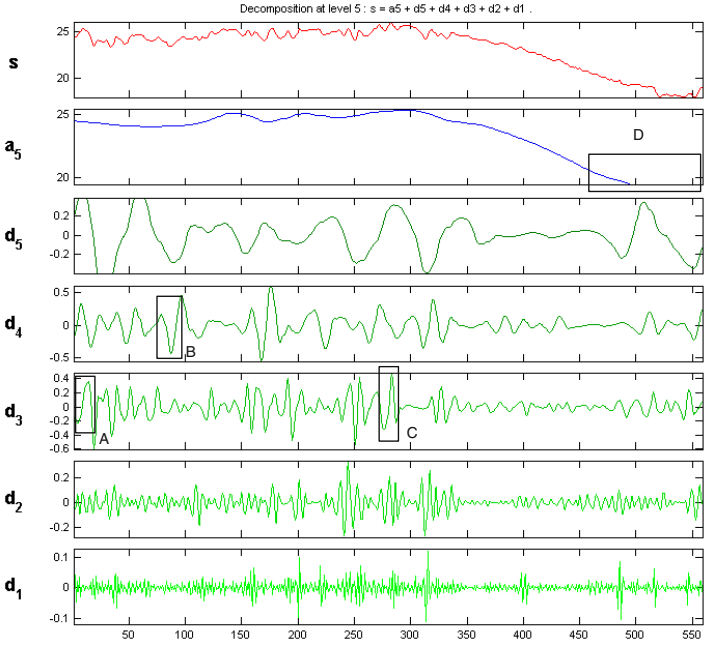


Fig. 4. Wavelet decomposition

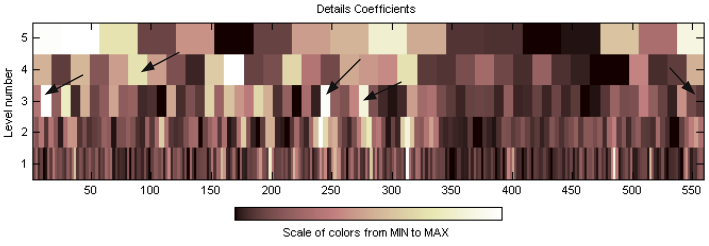


Fig. 5. Scalogram

level a_5 at interval $140-300$), which corresponds to location $179.9°E-132.6°W$ when the temperature is between $26-24°C$. Sharp SST changes appear in the detailed part of the decomposition at different scales. For example, local outliers A and C are clearly detected at scale 3 and B in scale 4. At each scale, peaks appearing in detailed part of the decomposition are sharp SST; changes for short time duration are also good candidates of local outliers. Again, global outlier location D is detected in the approximation level at scale 5.

Neighborhood difference is clearly observed in the scalogram analysis. Brighter region gives higher similarity of the feature with the wavelet and darker region gives greater dissimilarity. At a particular scale, a brighter region surrounded by darker regions in color intensity is a good candidate of outlier. Some outliers are shown in Fig. 5 using arrows which are detected as outliers (Fig. 1) at the same location for particular scale.

3.2 Parallel Algorithm

In this section, we show the scalability of the proposed PWT algorithm over the sequential algorithm for outlier detection. We validated the scalability of PWT for a dataset of mean temperature collected over a time period (1991-1999) by NOAA NCDC daily GLOBALSOD: Global daily WMO weather station data.

We implemented both sequential and parallel wavelet transform algorithms. The block data distribution approach in our parallel algorithm gives the first $\lg \frac{N}{P} - 1$ stages computation intensive and the rest $\lg N - (\lg \frac{N}{P} - 1)$ stages communication intensive. For the first $(\lg \frac{N}{P} - 1)$ stages, input data required to compute the coefficients are local in the processor except two input data located in neighboring processors. As the communication is fixed and small, these stages give scalability with the increase of data volume. At $\lg \frac{N}{P}$-th stage, each processor will have one value to compute and communication will be 6. As the number of data is reduced by 2 at each level, after $(\lg \frac{N}{P} + 1)$ stages, half of the processors will be idle and others will have one value to compute and number of communications is still 6. Therefore, these communication intensive stages will not contribute much to scalability. Here, scalability can be ensured in two ways. First, the outlier mining algorithm analyzes data up-to a particular number of scales (level). Spatial outliers are in general local outliers based on

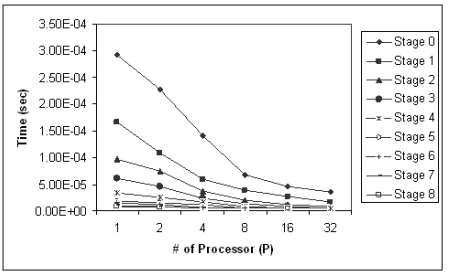


Fig. 6. Execution time in computation intensive stage

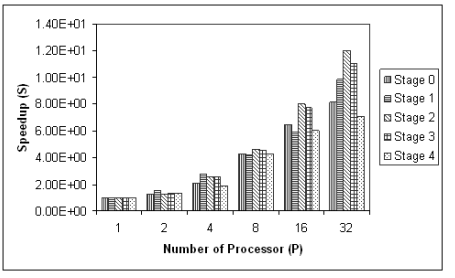


Fig. 7. Speedup for $N = 1024$

the neighborhood resolution, hence smaller scales are adequate to capture the high frequency feature change in the data. The last few communication intensive stages generate very few coefficients. They are used to detect the trend in data, and in most cases are insignificant for the outlier mining application. Secondly, with huge dataset, the time efficiency achieved at computation intensive stages will be more to overlook the time required in the communication intensive stages.

Table 1. Timing analysis for communication intensive stages

Stage	Required time (sec) in processors				
	2	4	8	16	32
5					$1.04E-05$
6				$1.20E-05$	$2.08E-05$
7			$1.52E-05$	$1.84E-05$	$8.00E-07$
8		$2.08E-05$	$1.60E-06$	$8.00E-07$	$7.99E-07$
9	$1.28E-05$	$2.40E-06$	$8.00E-07$	$8.00E-07$	$8.00E-07$

Fig. 6 shows the execution time for 1024 discrete data points with different number of processors for the first $\lg \frac{N}{P}$ stages. From the plot, we find that with the increase in processors, the required time is decreasing. We also conducted the time analysis for communication (Table 1) intensive stages by varying the number of processors. In most cases, the spent time in a processor is approximately $8.00E-07$ sec and sometimes deviation is also observed, depending upon the speed of the network and distribution of nodes in the cluster. To realize the impact of communication latency on total time, we can compare the execution time required in computation and communication intensive stages. With 16 processors, at stage 1, the time required for 32 coefficients computation and 2 communications is $2.80E-05$ sec, whereas at stage 7 the required time for 1 coefficient computation and 6 communications is $1.84E-05$ sec.

As explained in Section 2.2, the number of communication and computation intensive stages depend on $\lg P$ and $\lg \frac{N}{P}$, respectively. Therefore, with higher value of N, the impact of $\lg P$ stages is tolerable, otherwise a specific value of P can only give scalability for small value of N in the total time for communication

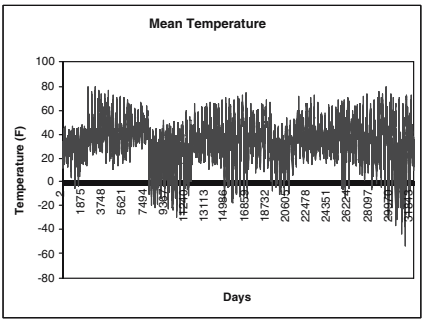


Fig. 8. Mean temperature

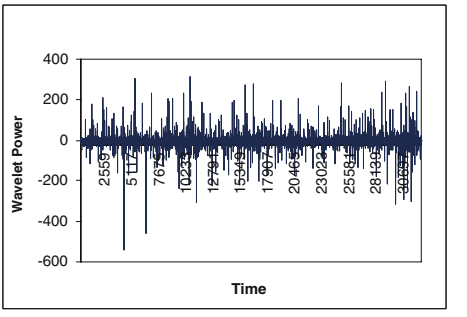


Fig. 9. Wavelet coefficients

and computation intensive stages. For large complex spatial datasets, the computational cost will be higher and communication cost will be lower compared to the computational cost. In such scenario, increasing the number of processors will help in ensuring more scalability. Yet a specific number of processors for a data size will give the best result. In Fig. 7, we observe speedup increase at each stage with the increase in processors. This can be explained from Fig. **??**, where we have shown the computation for 32 data points with 4 processors. For the first 4 stages, each processor sends two border data to its predecessor processor and receives two data (which are border data too) from the neighboring processor. The communication cost required is the sum of one send and one receive. Each data point needs 4 multiplications and 1 summation, and at level i, the number of data points computed is $\frac{N}{P\times 2^i}$. If t_{mul} and t_{add} denote the computational time for one multiplication and one addition, respectively, the time required to compute one data point is $4\times t_{mul}+t_{add} = t_{comp}$. Therefore, the total computational time required in a processor at level i is $\frac{N}{P\times 2^i} \times t_{comp}$. If t_{comm} denotes communication time, each processor spends $2 \times t_{comm}$ for communication. Hence, the total time incurred in a processor is $\frac{N}{P\times 2^i} \times t_{comp} + 2 \times t_{comm}$. With the increase of P, the first part (computation) of the above expression is decreased, but communication time will remain the same and the newly added processors will add more communication. Each stage starts with the completion of the previous stage and any delay in the interprocess communication time of the previous stage delays the start of next stage, hence the total time (for all stages) is increased. But for large datasets, the computational cost is higher than communication. Hence, the time efficiency achieved from computation intensive stages is more to neglect any anomaly or a constant increase of communication and will give more scalability with increase in processors.

We used parallel wavelet transform algorithm to calculate wavelet power for both datasets. Fig. 9 plots the wavelet power of the mean temperature (Fig. 8). Our db4 parallel wavelet algorithm decomposed the signal up-to 15 levels. The wavelet coefficients at the 15^{th} level accumulate the detailed and approximate

parts of the decomposition. In Fig. 9, the spikes are anomalies from the regular temperatures. They are deviated from the general trend of temperature pattern and are treated as outliers. If we watch carefully, we will see that in most cases, more than one spikes are grouped in one place, thus forming a cluster. These small groups are forming our suspected outliers, meaning the temperature deviation for few days.

4 Conclusions

In this paper, we applied wavelet analysis for outlier mining to take the advantage of multi-scale capability and multi-resolution feature. Zhao *et al* [14] used a wavelet based approach and they used Morlet and Mexican Hat mother wavelet. Whereas, Daubechies gives better frequency resolution and we used discrete fast wavelet transform which is more suitable to implement. Further, parallelization gives speedup when the spatial data objects are very large. So far, we deal with two features (temperature, latitude or time) in the weather data; considering its multi-dimensional attribute domain (e.g., vapor, air pressure, etc.) at the same time poses a challenging issue for meteorological forecasting. We also plan to improve the parallel algorithm for different load distributions, granularity, and data locality issues to reduce communication latency. We will test the outlier mining algorithm using other statistical (e.g., iterative-Z value) and visualization (e.g., variogram clouds, pocket plots, scatter plot) approaches.

References

1. Barua, S., Alhajj, R.: High Performance Computing for Spatial Outliers Detection Using Parallel Wavelet Transform. Intelligent Data Analysis (in press)
2. Barua, S., Alhajj, R.: A Parallel Multi-scale Region Outlier Mining Algorithm for Meteorological Data. In: Proc. of ACM GIS (2007)
3. Birant, D., Kut, A.: Spatio-temporal outlier detection in large databases. Journal of Computing and Information Technology 14(4), 291–298 (2006)
4. Cheng, T., Li, Z.: A multiscale approach for spatio-temporal outlier detection. Transactions in GIS 10(2), 253–263 (2006)
5. Edwin, M.K., Raymond, T.N.: A unified notion of outliers: Properties and computation. In: Proc. of ACM-KDD, pp. 219–222 (1997)
6. Hung, E., Cheung, D.: Parallel algorithm for mining outliers in large database. In: Proc. of IDC (1999)
7. Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: Proc. of VLDB, pp. 392–403 (1998)
8. Ramachandran, R., Rushing, J., Conover, H., Graves, S., Keiser, K.: Flexible framework for mining meteorological data. In: Proc. of IIPS for Meteorology, Oceanography, and Hydrology (February 2003)
9. Ramaswamy, S., Alto, P., Rastogi, R., Shim, K.: Efficient algorithms for mining outliers from large data sets. In: Proc. of ACM SIGMOD, pp. 427–438 (2000)
10. Shekhar, S., Chawla, S., Ravada, S., Fetterer, A., Liu, X., Lu, C.: Spatial databases-accomplishments and research needs. IEEE TKDE 11(1), 45–55 (1999)

11. Shekhar, S., Lu, C.-T., Zhang, P.: A unified approach to detecting spatial outliers. GeoInformatica 7(2) (2003)
12. Barnett, T.L.V.: Outliers in Statistical Data. John Wiley, New York (1994)
13. Yu, D., Sheikholeslami, G., Zhang, A.: Findout: finding outliers in very large datasets. Knowl. Inf. Syst. 4(4), 387–412 (2002)
14. Zhao, J., Lu, C., Kou, Y.: Detecting region outliers in meteorological data. In: Proc. of ACM International Symposium on Advances in GIS, pp. 49–55 (2003)

A Tool for Web Usage Mining

Jose M. Domenech[1] and Javier Lorenzo[2]

[1] Hospital Juan Carlos I
Real del Castillo 152 - 3571 Las Palmas - Spain
jdomcab@gobiernodecanarias.org
[2] Inst. of Intelligent Systems and Num. Applic. in Engineering
Univ. of Las Palmas
Campus Univ. de Tafira - 35017 Las Palmas - Spain
jlorenzo@iusiani.ulpgc.es

Abstract. This paper presents a tool for web usage mining. The aim is centered on providing a tool that facilitates the mining process rather than implement elaborated algorithms and techniques. The tool covers different phases of the CRISP-DM methodology as data preparation, data selection, modeling and evaluation. The algorithms used in the modeling phase are those implemented in the Weka project. The tool has been tested in a web site to find access and navigation patterns.

1 Introduction

Discovering knowledge from large databases has received great attention during the last decade being the data mining the main tool to make it [1]. The world wide web has been considered as the largest repository of information but it lacks of a well defined structure. Thus the world wide web is a good environment to make data mining receiving the name of Web Mining [2,3].

Web mining can be divided into three main topics: Content Mining, Structure Mining and Usage Mining. This work is focused on Web Usage Mining (WUM) that has been defined as "the application of data mining techniques to discover usage patterns from Web data" [4]. Web usage mining can provide patterns of usage to the organizations in order to obtain customer profiles and therefore they can make easier the website browsing or present specific products/pages. The latter has a great interest for businesses because it can increase the sales if they offer only appealing products to the customers although as pointed out Anand (Anand et al, 2004), it is difficult to present a convincing case for Return on Investment. The success of data mining applications, as many other applications, depend on the development of a standard. CRISP-DM, (Standard Cross-Industry Process for Data Mining) (CRISP-DM, 2000) is a consortium of companies that has defined and validated a data mining process that can be used into different data mining projects as web usage mining. The life cycle of a data mining project is defined by CRISP-DM into 6 stages: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment.

The Business Understanding phase is highly connected with the problem to be solved because they defined the business objectives of the application. The last

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 695–704, 2007.

one, Deployment, is not easy to make automatically because each organization has its own information processing management. For the rest of stages a tool can be designed in order to facilitate the work of web usage mining practitioners and reduce the development of new applications.

In this work we implement the WEBMINER architecture [5] which divides the WUM process into three main parts: preprocessing, pattern discovery and pattern analysis. This three parts corresponds to the data preparation, modeling and evaluation of the CRISP-DM model.

In this paper we present a tool to facilitate the Web Usage Mining based on the WEBMINER architecture. The tool is conceived as a framework where different techniques can be used in each stage facilitating in this way the experimentation and thus eliminating the need of programming the whole application when we are interested in studying the effect of a new method in the mining process. The architecture of the tool is shown in Figure 1 and the different elements that makes up it will be described. Thus, the paper is organized as follows. Section 2 will describe the data preprocessing. In sections 3 and 5 different approaches to user session and transactions identification will be presented. Finally in sections 6 and 7 the models to be generate and the results are presented.

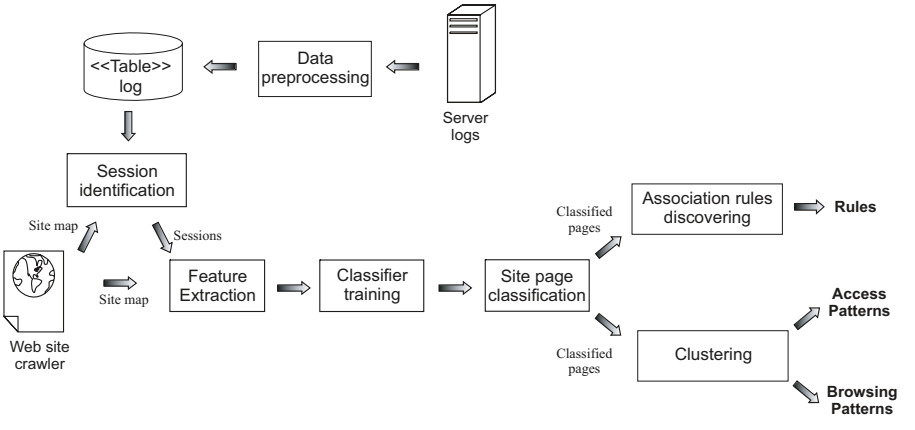


Fig. 1. WUM tool architecture

2 Web Log Processing

Data source for Web Usage Mining come from different sources as proxy, web log files, web site structure and even from sniffer packet logs. Normally, the most widely used sources are the web log files. These files record the user accesses to the site and there exists several formats: NCSA (Common Log Format), W3C Extended, SunTM ONE Web Server (iPlanet), IBM Tivoli Access Manager WebSEAL or WebSphere Application Server Logs. The most of the web servers

record the access using an extension of the CLF (ECLF). In ECLF basically the recorded information for each access is:

- *remote host:* Remote hostname. (or IP address number if DNS hostname is not available or was not provided)
- *rfc931:* The remote login name of the user. (If not available a minus sign is typically placed in the field)
- *authuser:* The username as which the user has authenticated himself. This is available when using password protected WWW pages. (If not available a minus sign is typically placed in the field)
- *date:* Date and time of the request.
- *request:* The request line exactly as it came from the client. (i.e., the file name, and the method used to retrieve it [typically GET])
- *status:* The HTTP response code returned to the client. Indicates whether or not the file was successfully retrieved, and if not, what error message was returned.
- *bytes:* The number of bytes transferred.
- *referer:* The url the client was on before requesting your url. (If it could not be determined a minus sign will be placed in this field)
- *user agent:* The software the client claims to be using. (If it could not be determined a minus sign will be placed in this field)

As said before, web server logs record all the user accesses including for each visited page all the elements that composed it as gif images, styles or scripts. Other entries in the log refers to fail requests to the server as "404 Error: Object not found". So a first phase in data preparation consists of filtering the log entries removing all useless entries. Others entries in the web log that must be removed, are those that correspond to search robots because they do not corresponds to a "true" user. To filter these entries it can be used the plain text file `Robot.txt`, the list of known search robots `www.robotstxt.org/wc/active/all.txt` and we have introduced an heuristic that is to filter those very quick consecutive requests because a characteristic of search robots is the short delay between page requests. So with a threshold of 2 seconds between two consecutive requests the entries that corresponds to robots can be eliminated.

The structure of the site has been used as another data source. This structure is obtained with a web crawler starting from the root, so all the pages that can be reached from the root will composed the structure of it. For non static sites the structure must be introduced by hand.

3 User Session Identification

Once the web log file is processed and all the irrelevant entries has been removed, it is necessary to identify the users that visit to the site. The visits are concurrent so in the log file the entries of different users are interlaced what makes us process it to collect the entries that belong to the same user.

A first approach to identify a user is to use the IP address and assign all the entries with the same IP to the same user. This approach exhibits some

drawbacks. Some users access to internet through a proxy so many users will share the same IP. In other cases the same user has different IP because it has a dynamic IP configuration in its ISP. In order to minimize these effects some heuristics has been applied. A first heuristic is to detect changes in the browser or in the operative system fields of the entries that come from the same IP. Another heuristic makes use of the referer field and the map of the site obtained with the site crawler mentioned previously. Thus if a page is not directly linked to the pages previously visited by the user, it is an evidence that another user share the same IP and browser. With the explained heuristics we will get false positive, that is to consider only one user when actually are different users.

After identifying the users that have visited the site, the next phase is to obtain the user sessions. A session is made up of all the visited pages by a user. The technique is based on establishing a time threshold, so if two accesses take more than the fixed time threshold, it is considered as a new session [6,7]. Many commercial products establish a threshold of 30 minutes. Catledge and Pitkow [8] define this threshold in 25.5 minutes based on empirical studies.

4 Web Page Classification

After data cleaning, the next stage in the data preparation phase is to compute the features of each page in the site. The following features has been defined:

- *Size:* Size of the page in bytes.
- *Num. incoming links.*
- *Num. outcoming links.*
- *Frequency:* Number of times the page was requested in a period of time.
- *Source:* Number of times the page is the starting point in a session.
- *Similarity:* Similarity of a page with its sibling pages based on a tbd computation.
- *Depth:* Average depth of the sibling pages. The depth of a pages is measured as the number of '/' in the URL.

From the previous features it can be obtained a model for diferent pages which avoid to the webmaster to annotate each of the page in the site. In this work we have defined the following pages of interest:

- *Home page:* It is the first visited page by the users.
- *Content page:* It contains a part of the site information.
- *Auxiliary page:* Users can use this page to visit other pages in the site.
- *Reference page:* Explain a concept or it has references
- *Personal page:* It contains biographic information of the organization staff.

To avoid the computational cost of training a classifier with the whole set of features, a previous feature selection stage is made. The initial feature set is filtered using the GD measure [9], which is based on information theory concepts, in order to select the most informative features. This measure allows to rank the

features according to the relevance with the concept and it also detects redundant features that can be removed from the initial feature set.

In a small web site, pages can be manually tagged as home page, content page and so on, but in a medium or large web site this is not affordable. Therefore it is necessary an automatic or semi-automatic method to tag the pages. In this proposal a phase of page classification is include (Figure 1) based on a learned model for the different categories of pages and using the features defined above. Hwanjo et al. [10] propose to use SVM with positive samples to classify web pages. Xu et al. [11] also introduce the SVM to deal with the heterogeneous data that appear in a web page as link, plain text, title page or anchor text. Holden and Freitas [12] make use of the Ant Colony paradigm to find a set of rules that classify the web pages into several categories. The study of complex web page classification algorithms is out of the scope of this paper so two well known learning methods have been included: naive-bayes and C4.5.

In this tool, a supervised learning stage has been included. The user selects and manually tags a set of pages that makes up the initial training set. With this initial training set, the learning process is launched and the results are tested by the user. If there are bad classified pages, the user can introduce them into the learning set with the correct tag. After some cycles, a correct model is obtained and the pages of the site are classified.

5 Transaction Identification

A transaction is defined as a set of homogeneous pages that have been visited in a user session. Each user session can be considered as only one transaction composed of all the visited pages or it can divided into a smaller set of visited pages. The transaction identification process is based on a split and merge process in order to look for a suitable set of transactions that can be used in a data mining task.

Formally, a transaction is composed of an IP address, a user identification and a set of visited pages which are identified by its URL and access time.

$$t =< ip_t, uid_t, \{(l_1^t.url, l_1^t.time), ..., (l_m^t.url, l_m^t.time)\} >$$

$$For\ 1 \leq k \leq m,\ l_k^t \in L,\ l_k^t.ip = ip_t,\ l_k^t.uid = uid_t \tag{1}$$

To realize the split stage in the transaction identification there are different strategies.

Transaction Identification by Reference Length. This strategy, proposed by Cooley et. al. [2], is based on the assumption that the time that a user spends in an auxiliary page is lower than a content page. Obtaining a time t by a maximum likelihood estimation and defining a threshold C, the pages are added to the transaction if they are considered auxiliary-content:

$$1 \leq k \leq (m-1) : l_k^{trl} length \leq C\ and\ k = m : l_k^{trl} length > C \tag{2}$$

While for only content pages transactions:

$$1 \leq k \leq m : l_k^{trl} length > C \tag{3}$$

Transaction identification by Maximum Forward Reference. This strategy is based on the idea proposed by Chen et al. [13]. A transaction is considered as the set of pages from the first visited page until the previous page where the user does a back reference. A back reference appears when the user accesses again to a previously visited page in the current session, while a forward reference is to access to a page not previously visited in the current session. So the maximum forward reference are the content pages and the path to the maximum reference is composed of index pages.

Transaction Identification by Time Window. This strategy divides a user session into time intervals lower than a fixed threshold. In this strategy the last visited page normally does not correspond to a content page unlike the previous strategy. If W is the size of the time window, the accesses that are included to the transaction (1) are those that fulfill:

$$l_m^t time - l_1^t time \leq W \tag{4}$$

This strategy is normally combined with the previous ones.

6 Model Generation

To characterize the visitors of the web site, it is interesting to detect the access patterns, that is, what type of pages are visited and also navigation patterns, that is, how the visitors browse the pages in the web site. Both patterns are of interest because they can help the web site designer to improve the usability or visibility of the site. To get these patterns a clustering stage is introduced into the tool and although many works have been proposed to deal with this problem [14,15,16], in the tool three well know methods have been used: Expectation Maximization, K-means and Cobweb. As input to the previous methods, both the identified sessions and the transactions are used.

Another information that is useful for the web site designer is to know if there exists any unusual relation among the pages that are visited by users. This information can be easily extracted from the transactions and user sessions by means of an association rule discovering module. The Apriori method proposed by Agrawal [17] has been used.

7 Experiments

The tool was implemented in Java and the learning methods were the ones implemented in Weka [18] and by now we are only focused in the development of the

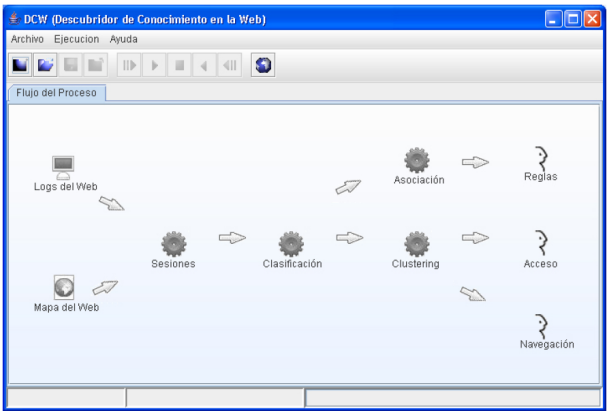


Fig. 2. DCW tool

framework it will allow us to introduce new learning methods. The appearance of the tool is shown in Figure 2.

To test the approach we select a log file corresponding to a week of accesses to the site of the Department of Computer Science (DIS) of the Univ. of Las Palmas de Gran Canaria (`http://www.dis.ulpgc.es`). The log file has 478512 entries that after the preprocessing phase (Sec. 2) it is reduced to 25538 entries.

Fixing a time threshold for session identification (Sec. 3) to 30 minutes, 9460 sessions were identified being the main page of the DIS the most visited pages with 1571 accesses. After the session identification the next stage is to train the classifier to tag the pages. In this experiment the pages where divided into two categories: content and auxiliary. The C4.5 algorithm was used to do induce a decision tree model and an accuracy of 85% was achieved.

Once the pages of the site are classified into auxiliary or content category, the pattern extraction is carried out. To get the access pattern of the visitors a clustering phase with EM algorithm is done. The results are the shown in Table 1. Two different clusters are obtained with correspond to users that visit mainly content pages while the other cluster represents the visitors that browse auxiliary pages. The first cluster could correspond to students and staff while the second one could correspond to "curious" visitors because they only browse auxiliary pages.

Table 1. Access patterns results with EM clustering

	Content pages	Auxiliary pages
Cluster 0	1	0.0323
Prob.=0.7012	D = 0.4463	D = 0.1768
Cluster 1	0	1
Prob.=0.2988	D = 0.4463	D = 0.4574

Log likelihood: -0.26076

To get the access patterns, only sessions of 3, 4 o 5 accesses (pages visited) are considered. Table 2 shows the clusters obtained for 3 accesses sessions. The largest cluster corresponds to sessions that end up in auxiliary pages which means that the user abandons the site before reaching a page that gives useful information.

Table 2. Navigation patterns for 3 accesses sessions

	access 0	**access 1**	**access 2**
Cluster 0	Auxiliary page	Auxiliary page	Auxiliary page
Prob.=0.6825	350(376.03)	374.97 (376.03)	375.03 (376.03)
Cluster 1	Content page	Content page	Content page
Prob.=0.3175	174.97(175.97)	174.97(175.97)	174.97(175.97)

Log likelihood: -0.8412

Table 3 shows the results for the access patterns of session with 4 accesses and here it can be noted that the two largest clusters correspond to sessions that finish in content pages and only a small amount of sessions end up in auxiliary pages which can imply that the visitor does not find the information that was looking for.

Table 3. Navigation patterns for 4 accesses sessions

	access 0	**access 1**	**access 2**	**access 3**
Cluster 0	Content page	Content page	Content page	Content page
Prob.=0.676	92.95(93.95)	92.9(93.95)	87.82(93.95)	86.71(93.95)
Cluster 1	Auxiliary page	Auxiliary page	Auxiliary page	Auxiliary page
Prob.=0.2601	34.47(36.51)	35.49(36.51)	35.51(36.51)	35.49(36.51)
Cluster 2	Auxiliary page	Auxiliary page	Auxiliary page	Content page
Prob.=0.0639	10.01(11.54)	8.46(11.54)	10.35(11.54)	9.27(11.54)

Log likelihood: -1.21079

Table 4 shows the association rules that were obtained with Apriori algorithms. The rules do not contribute to generate new knowledge because they are very obvious. For example the first rule expresses that if the second visited page is the studies "Informatic Engineering", the first page was the root of the site.

As the aim of this work is to present the framework for a WUM tool, therefore a comparative of the results with other techniques has not been carried out because they can be found in the literature. Comparing with other open source tools, we have found that the most similar is WUMprep [19] which only cover part of the Data Preparation stage and unlike DCW that has a GUI, WUMprep it is based on Perl script. In relation to the model generation and validation there are two well-know tools as Weka [18] and RapidMiner [20]. They are oriented to data mining in general and the previous stage of web log cleaning must be done with another tools.

Table 4. Association rules

Rules	Support	Confidence
access1=/subject/index.asp?studies=ITIS =¿ access0=/	16	1
access1=/subject/index.asp?studies=II =¿ access0=/	14	1
access2=/staff/ =¿ access0=/	16	1
access2=/student/ =¿ access0=/	15	1

8 Conclusions

In this work a tool for Web Usage Mining has been presented. It allows to realize all phases to get access and navigation patterns and also association rules. The implementation was done in Java and making use of the Weka inducers which allow to test new model induction algorithms. To test the tool, some experiments were carried out with a log file of more than 700.000 entries and they reveal some behaviors of the visitors that the designer of the web do not know and it can help them to redesign the web site to offer a better service to the students and staff of the Department of Computer Science of the ULPGC.

Future work is twofold. On the one hand, some elements of the proposed tool needs to be improved to tackle for example with dynamics web sites. One the other hand, other methods can be tested in the classification and clustering phases. In the page classification phase the computation of new features and the use of SVM as classifier.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Education and Science and FEDER funds under research project TIN2004-07087. Thanks to Miguel Garcia from La Laguna University for his implementation in Java of the GD Measure.

References

1. Hand, D.J., Mannila, H., Smyth, P.: Principles of Data Mining. The MIT Press, Cambridge (2001)
2. Cooley, R., Mobasher, B., Srivastava, J.: Data preparation for mining world wide web browsing patterns. Journal of Knowledge and Information Systems 1, 5–32 (1999)
3. Chakrabarti, S.: Mining the Web.Discovering Knowledge from Hypertext Data. Morgan-Kaufmann Publishers, San Francisco (2003)
4. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.N.: Web usage mining: Discovery and applications of usage patterns from web data. SIGKDD Explorations 1, 12–23 (2000)
5. Cooley, R., Srivastava, J., Mobasher, B.: Web mining: Information and pattern discovery on the world wide web. In: ICTAI 1997. Proc. of the 9th IEEE International Conference on Tools with Artificial Intellegene (1997)

6. Berendt, B., Mobasher, B., Nakagawa, M., Spiliopoulou, M.: The impact of site structure and user environment on session reconstruction in web usage analysis. In: Zaïane, O.R., Srivastava, J., Spiliopoulou, M., Masand, B. (eds.) WEBKDD 2002. LNCS (LNAI), vol. 2703, pp. 159–179. Springer, Heidelberg (2003)
7. Spiliopoulou, M., Mobasher, B., Berendt, B., Nakagawa, M.: A framework for the evaluation of session reconstruction heuristics in web usage analysis. INFORMS Journal on Computing 15, 171–190 (2003)
8. Catledge, L.D., Pitkow, J.E.: Characterizing browsing strategies in the World-Wide Web. Computer Networks and ISDN Systems 27, 1065–1073 (1995)
9. Lorenzo, J., Hernández, M., Méndez, J.: Detection of interdependences in attribute selection. In: Żytkow, J.M. (ed.) PKDD 1998. LNCS, vol. 1510, pp. 212–220. Springer, Heidelberg (1998)
10. Yu, H., Han, J., Chang, K.C.C.: Pebl: Web page classification without negative examples. IEEE Transactions on Knowledge and Data Engineering 16, 70–81 (2004)
11. Xu, Z., King, I., Lyu, M.R.: Web page classification with heterogeneous data fusion. In: WWW2007. Proceedings of the Sixteenth International World Wide Web Conference, Alberta, Canada, pp. 1171–1172 (2007)
12. Holden, N., Freitas, A.: Web page classification with an ant colony algorithm. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN VIII. LNCS, vol. 3242, pp. 1092–1102. Springer, Heidelberg (2004)
13. Chen, M.S., Park, J.S., Yu, P.S.: Data mining for path traversal patterns in a Web environment. In: Proceedings of the 16th International Conference on Distributed Computing Systems (1996)
14. Xiao, J., Zhang, Y.: Clustering of web users using session-based similarity measures. In: ICCNMC 2001. 2001 International Conference on Computer Networks and Mobile Computing, pp. 223–228 (2001)
15. Bianco, A., Mardente, G., Mellia, M., Munafo, M., Muscariello, L.: Web user session characterization via clustering techniques. In: GLOBECOM 2005. Global Telecommunications Conference, IEEE, Los Alamitos (2005)
16. Chen, L., Bhowmick, S.S., Li, J.: Cowes: Clustering web users based on historical web sessions. In: 11th International Conference on Database Systems for Advanced Applications, Singapore (2006)
17. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: 20th Int. Conference on Very Large Data Bases, pp. 487–499 (1994)
18. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
19. Pohle, C., Spiliopoulou, M.: Building and exploiting ad hoc concept hierarchies for web log analysis. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) DaWaK 2002. LNCS, vol. 2454, Springer, Heidelberg (2002)
20. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale (now: Rapidminer): Rapid prototyping for complex data mining tasks. In: KDD 2006. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2006)

An Algorithm to Mine General Association Rules from Tabular Data

Siyamand Ayubi[1], Maybin Muyeba[2], and John Keane[3]

[1] Faculty of Engineering
[2] Liverpool Hope University, School of Computing, Liverpool, UK
[3] University of Manchester, School of Computer Science, Manchester, UK
s.ayubi@gmail.com, muyebam@hope.ac.uk,
john.keane@manchester.ac.uk

Abstract. Mining association rules is a major technique within data mining and has many applications. Most methods for mining association rules from tabular data mine simple rules which only represent equality in their items. Limiting the operator only to "=" results in many interesting frequent patterns that may exist not being identified. It is obvious that where there is an order between objects, greater than or less than a value is as important as equality. This motivates extension, from simple equality, to a more general set of operators. We address the problem of mining general association rules in tabular data where rules can have all operators $\{\leq, >, \neq, =\}$ in their antecedent part. The proposed algorithm, Mining General Rules (MGR), is applicable to datasets with discrete-ordered attributes and on quantitative discretized attributes. The proposed algorithm stores candidate general itemsets in a tree structure in such a way that supports of complex itemsets can be recursively computed from supports of simpler itemsets. The algorithm is shown to have benefits in terms of time complexity, memory management and has great potential for parallelization.

Keywords: data mining, general association rules, tabular data, equality operators.

1 Introduction

Association rule (AR) mining [1] has been traditionally applied to datasets of sales transactions (referred to as market basket data). A transaction 'T' is a set of items and contains an itemset 'I' if I $\subseteq$ T. If 'I' has k members, then 'I' is called a k_itemset. An AR is an implication $X \rightarrow Y$ where X and Y are itemsets with no items in common i.e. $X \cap Y = \varnothing$. The intuitive meaning of such a rule is that the transactions (or tuples) that contain X also contain Y. The rule $X \rightarrow Y$ holds with confidence c if c% of transactions that contain X also contain Y. The rule $X \rightarrow Y$ has a support s if s% of the transactions in the database contain X $\cup$ Y. Given a database, the problem of mining ARs is to generate all rules that have support and confidence greater than the

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 705–717, 2007.

user-specified minimum thresholds, min-Support and min-Confidence. There are many algorithms to mine association rules from transactional data [6, 7]. The AR technique has also been applied to tabular datasets [2, 3, 5, 11]. The notation for an item is redefined in tabular datasets. Henceforth an item is a triple (a, Θ, v) where "a" is an attribute, "v" is the value of "a" and Θ is the operator between "a" and "v". An example of an AR in tabular data is as follows:

$(A_1 = 2)$ and $(A_2 = 3)$ and $(A_4 = 5) \Rightarrow A_6 = 1$ support = 10%, Confidence = 60%

where $A_{1,...,}A_6$ are attributes with equality operator "=" and referred to as *simple rules* [2, 9]. In some cases, simple rules are unable to show all hidden patterns of data. In situations where there are orders between values of attributes, greater than or less than a value is as important as equality. Simple rules have difficulties in extracting such patterns, their drawback being in dealing with quantitative attributes. Popularly, quantitative attributes can be discretized by partitioning domain values to base intervals, where the difficulty is selecting the number of base intervals. Too many intervals for a quantitative attribute means low support for single intervals. Hence some rules involving these attributes may not be found, on the other hand, partitioning values into too few intervals causes information to be lost. Some rules have maximum confidence only when some of the items in their antecedent part have small intervals. These problems can be solved to some extent by using $\{\leq, >, \neq, =\}$ operators in the items. The MinSup problem does not matter for these operators as the number of intervals has no effect on operators $\{\leq, >, \neq\}$. If the operators of items in a rule belong to $\{\leq, >, \neq, =\}$, then the rule is called a *general rule* [2, 9]. An example of such a rule is as follows:

$(A_1 > 2) and (A_2 \neq 3) and (A_4 = 5) \rightarrow (A_6 = 1)$ Support = 10%, Confidence = 60%

In this paper, we propose an algorithm, MGR (Mining General Rules), to mine general rules. The number of general itemsets is exponentially higher than the number of simple itemsets and it makes mining them too difficult. Thus, MGR mines general itemsets from simple itemsets by a recursive computation of simpler itemsets, all stored in a tree data structure. This feature systematically enables benefits in terms of time complexity, memory management and great potential for parallelization.

The paper is organized as follows: background and related work is given in section 2; section 3 presents needed terminology and notation; section 4 shows calculation of supports of general itemsets from simple ones; section 5 presents the MGR algorithm; experimental results are given in Section 6; and Section 7 considers further work and presents conclusions.

2 Background and Related Work

In this paper we are interested in a type of generalization introduced in [2]. This type of generalization extends the traditional equality operator used in normal association rules to an operator set $\{\leq, >, \neq, =\}$. Its main characteristic is the high number of itemsets generated in regard to normal association rules. Hsu et al. [9] proposed an

algorithm for mining such general rules from transactional data by extending algorithms for mining simple itemsets. This approach has poor performance on tabular data because of the high number of general itemsets.

The main problem of mining association rules when applied to tabular datasets is dealing with quantitative attributes, which are usually partitioned into intervals and treated as discrete values. However, specifying the number of intervals is difficult in this approach. In [11], consideration of the combination of base intervals as new intervals and extracting the itemsets for all the intervals was proposed. The approach has an acceptable result but in most cases, its time complexity is high.

There are approaches that do not require discretization of quantitative attributes. Some extract types of rules that are different to formal association rules [10, 13]. The others do not try to mine all general itemsets but rather focus on finding the best intervals for a given rule [4]. The input to these algorithms is a rule that does not have any interval for its quantitative items. The outputs are the best intervals for quantitative attributes of the given rule. In situations where we are interested in optimizing one rule, these approaches are very useful. However, they cannot be applied to mine all general rules as they have to scan data for each rule.

There are some methods that define some criteria for interestingness of the general rules and mine just a subset of the rules that satisfy those criteria. The method in [14] defines the pc_optimal set as the set of the most interesting general association rules and tries to find an approximation of it. Despite the good results of the approach on some datasets, it is not guaranteed to find good approximations of the pc_optimal set. Further, it is not proved that the pc_optimal set contains the whole set of interesting rules.

3 Terminology and Notation

The following formal definitions are used to describe our approach and to prove related Lemmas.

Definition 1*(I):* *The set of all attributes of a table.*

We assume that attribute values are finite and discrete and there is an order between them. The finiteness assumption of attribute values is not restrictive because by partitioning infinite sets into subsets, we can easily convert infinite domains to finite domains. We use a_i for the ith attribute in I and $V_{i,j,}$ for jth value of ith attribute. As there is an order between the values of each attribute, $V_{i,1} < V_{i,2},\ldots, < V_{i,n}$.

Definition 2 ***(Item):*** *An item is a triple (a, Θ, v) where "a" is an attribute, "v" is a value of "a" and Θ is an operator between "a" and "v". An item is a simple item if its operator is "=". An item is a half general item, if the operator Θ is one of* $\{\leq, =\}$. *An item is a general item if the operator Θ is one of* $\{\leq, >, \neq, =\}$.

Note that the Θ cannot be $\{\geq, <\}$operators. As items have discrete values, we need not contain such operators e.g. ($i \geq j \Leftrightarrow i > j-1$) and ($i < j \Leftrightarrow i \leq j-1$).

Definition 3 ***(Itemset):*** *An itemset is the set of items (from definition 2). If every item in the itemset is a simple item, the itemset is a simple itemset. If every item in the itemset is a half general item, the itemset is a half general itemset. If every item in an itemset is a general item, the itemset is a general itemset.*

Definition 4 (t(X)): *t(X) is the set of IDs of records in a dataset which match the itemset X.*

4 Finding Supports of General Itemsets

Before we describe the MGR algorithm for mining general rules and itemsets, we first prove that we can obtain the supports of half general and general itemsets from simple and half general ones respectively. Lemma 2 explains the fact that the support of each half general itemset can be calculated from supports of simpler itemsets. Lemmas 3 and 4 explain the calculation of supports of general itemsets from half general ones.

Lemma 1. Let X be a half general itemset where its ith item has "$\leq$" as an operator and $V_{i,1}$ as a value. The support of X does not change if we convert the operator of the ith item to "=" and vice versa.

Proof: Because $V_{i,1}$ is the smallest value for the ith attribute, there are no records in the dataset with a smaller value than it, therefore all records in t(X) must have the value $V_{i,1}$ for the ith attribute. So changing operator "$\leq$" to "=" and vice versa does not change the support of the itemset.

Lemma 2. *Let X, Y and Z be half general itemsets that differ only in their ith item. They have the same attribute for the ith item but the operator of the ith item of X and Y is "$\leq$" and the operator of the ith item of Z is "=". The value of the ith item of X and Z is $V_{i,j}$ and the value of the ith item of Y is $V_{i,j-1}$, so the ith item in X and Z has one higher value than the ith item in Y. Then the supports of these itemsets have the following relationship:*

$$Sup(X) = Sup(Y) + Sup(Z)$$

Proof: Let a_i be the attribute of the ith item of X, Y and Z. As the ith item of X is $(a_i \leq V_{i,j})$, so for each $r \in t(X)$, we have $r(a_i) = V_{i,j}$ or $r(a_i) < V_{i,j}$ where $r(a_i)$ is the value of r for attribute a_i. Then t(X) can be partitioned into two subsets $t(X_1)$ and $t(X_2)$ according to the value of a_i such that the itemset X_1 contains $a_i = V_{i,j}$, and the itemset X_2 contains $a_i \leq V_{i,j-1}$ where the $V_{i,j-1}$ is the value before $V_{i,j}$. Therefore $t(X) = t(X_1) \cup t(X_2)$. As the operator of the ith item of Z is "=", then the itemset Z is equal to X_1. We can prove the same for itemsets Y and X_2. Therefore we have $t(X) = t(Y) \cup t(Z)$, $t(Y) \cap t(Z) = \phi$ and consequently we have $sup(X) = sup(Y) + sup(Z)$.

Table 1. A simple dataset with four Attributes $\{A_1, A_2, A_3, A_4\}$

Record Id	A1	A2	A3	A4
1	1	1	1	1
2	1	1	2	1
3	1	1	2	0
4	2	2	1	0
5	2	3	2	1
6	2	2	3	1
7	3	2	3	1
8	3	2	4	0
9	2	3	2	0
10	4	3	4	1
11	4	4	5	1

Example 1. *Suppose* we have a dataset as in Table 1, and the itemsets X, Y and Z have the following definitions:

$$X = (A_1 \le 2) and (A_2 \le 3) and (A_3 \le 2)$$
$$Y = (A_1 \le 2) and (A_2 \le 2) and (A_3 \le 2)$$
$$Z = (A_1 \le 2) and (A_2 = 3) and (A_3 \le 2)$$

Lemma 2 proves that *Sup(X) = Sup(Y) + Sup(Z).* As can be seen, the three itemsets differ in their second items. According to Lemma 2, we can partition X into the following itemsets:

$$X_1 = (A_1 \le 2) and (A_2 \le 2) and (A_3 \le 2)$$
$$X_2 = (A_1 \le 2) and (A_2 = 3) and (A_3 \le 2)$$

Hence $t(X_1) = \{1,2,3,4\}$, $\mathrm{Sup}(X_1) = 4$, $t(X_2) = \{5,9\}$, and $\mathrm{Sup}(X_2) = 2$.

As $t(X) = t(X_1) \cup t(X_2)$ then t(X) = {1,2,3,4,5,9} and $\mathrm{Sup}(X) = \mathrm{Sup}(X_1) + \mathrm{Sup}(X_2)$. As a result, $Y = X_1$ and $Z = X_2$, therefore Sup(X) =Sup(Y) + Sup(Z).

By using Lemma 1 and Lemma 2, the support of each half general itemset can be calculated from supports of simpler itemsets. In other words, if X is a half general itemset, then by applying Lemma 1 and Lemma 2 recursively, t(X) can be partitioned into $t(X_1)$,.., $t(X_n)$ where each X_i is a simple itemset.

Lemma 3. If X is a general itemset that does not have any item with an attribute a_i, then

$$Sup(X \cup (a_i > V_{i,j})) = Sup(X) - Sup(X \cup (a_i \le V_{i,j}))$$

Proof: Records of t(X) can be partitioned into two subsets such that $t(X) = t(X_1) \cup t(X_2)$, where $t(X_1)$ is the set of all records that have a value greater than $V_{i,j}$. ($t(X_1) = t(X \cup (ai > V_{i,j}))$) and $t(X_2)$ is the set of all records that has a value equal to or smaller than $V_{i,j}$, ($t(X_2) = t(X \cup (ai \le V_{i,j}))$). Then we have

$$t(X) = t(X \cup (a_i \le V_{i,j})) \cup t(X \cup (a_i > V_{i,j})) \text{ or}$$

$$Sup(X) = sup(X \cup (a_i > V_{i,j})) + Sup(X \cup (a_i \leq V_{i,j})) \text{ or}$$
$$Sup(X \cup (a_i > V_{i,j})) = Sup(X) - Sup(X \cup (a_i \leq V_{i,j}))$$

Lemma 4. *If X is a general itemset that does not have any item with the attribute a_i, then*

$$Sup(X \cup (a_i <> V_{i,j})) = Sup(X) - Sup(X \cup (a_i = V_{i,j}))$$

Proof: The proof is similar to that of Lemma 3.

Example 2. *If we extract itemsets X, Y and Z from Table 1, with the following definitions:*

$$X = (A_1 \leq 2) and (A_2 \leq 3) and (A_3 > 2)$$
$$Y = (A_1 \leq 2) and (A_2 \leq 3) and (A_3 \leq 2)$$
$$Z = (A_1 \leq 2) and (A_2 \leq 3)$$

then Lemma 3 proves that *Sup(X) = Sup(Z) - Sup(Y).* According to Table 1, we have:

$t(X)=\{6\}$, $t(Y)=\{12,3,4,5,9\}$, $t(Z)=\{1,2,3,4,5,6,9\}$
$t(X)=t(Z)-t(Y)$, so $Sup(X) = Sup(Z) - Sup(Y)$.

5 The MGR Algorithm

Although the number of general itemsets is exponentially higher than the number of simple itemsets, by applying the Lemmas of the previous section on itemsets in a systematic way, this enables the MGR algorithm to divide the problem into smaller ones and solve them more quickly. The main steps of the algorithm are as follows:

i. Mining simple itemsets using one of the existing algorithms.
ii. Mining half general itemsets from simple itemsets.
iii. Mining general itemsets from half general itemsets.
iv. Mining general rules from general itemsets.

The first step is achieved by using one of the existing methods for mining simple itemsets. The last step is similar to other association rule algorithms and we do not focus on it here. The main steps of the algorithm are steps 2 and 3. The MGR algorithm does these steps by applying Lemmas 1, 2, 3 and 4 on itemsets in a tree data structure called an MGR tree.

The MGR tree brings two benefits in mining general itemsets. Firstly, it facilitates finding itemsets in such a way that the Lemmas of section 4 can be applied more easily. Secondly, it breaks the problem of mining general itemsets from simpler problems. Before describing the MGR algorithm, we first explain the structure of the MGR tree.

The root of the MGR tree contains no data. The nodes of the first level of the tree are called *signatures*. All itemsets inside a signature have the same attributes. It is designed so in order to facilitate applying Lemma 2, as itemsets in Lemma 2 have the same attributes. Nodes of the second level of the tree are called Half General Itemsets

(HGI) nodes. Itemsets inside an HGI node have the same values for corresponding items, so each HGI node has just one simple itemset. Nodes of the last level of the tree are called GI (General Itemset) nodes. Each GI node has just one half general itemset. Itemsets inside each GI node can be created from its half general itemset by converting operators "=" and "≤" to operators "≠" and ">" respectively.

5.1 Mining Half General Itemsets

At this step of the algorithm, the extracted simple itemsets are partitioned into signatures. By defining a lexographical order between attributes and using the order between values of each attribute, the simple itemsets in each signature can be sorted, which is very important in finding itemsets. In the second level of the tree, there are HGI nodes which contain half general itemsets. Each simple itemset corresponds to an HGI node at this level of the tree. Half general itemsets of an HGI node have the same values for corresponding items, the only difference being the item operators. Mining half general itemsets in each signature begins from the first HGI node and up to the last one. The crucial issue here is that HGI node itemsets must be created in order according to figure 2. This enables the MGR algorithm to have random access to itemsets of an HGI node. Figure 1 shows the process of mining half general itemsets in each signature.

5.1.1 Illustrative Example of Mining Half General Itemsets

In this section, we illustrate the process of calculating the supports of half general itemsets. Suppose that itemsets of figure 3 are simple itemsets that belong to the signature $\{A_1, A_2\}$ and we want to calculate the supports of the half general itemsets of the signature. Figure 4 shows the HGI node of the first simple itemset. This HGI node is the first one that must be taken into account. The first itemset of this node is a simple itemset and its support is known. The supports of the other itemsets of the node are calculated by applying Lemma 1. Now let's consider the next HGI node which is shown in figure 5. The support of the first itemset is given (simple itemset). Support of itemset K_2 is calculated from the supports of itemsets K_1 and I_2 by applying Lemma 1. Support of itemset K_3 is calculated by applying Lemma 1. The same process can be done for other HGI nodes. In fact, we use the itemsets of the previous HGI nodes to calculate supports of an HGI node. If one of the required itemset does not exist, the process will be repeated to calculate its support. We suppose that absent simple itemsets have zero supports.

5.2 Mining General Itemsets

The process of mining general itemsets from half general itemsets is similar to the process of mining half general itemsets from simple ones. For each extracted half general itemset, a GI node (General Itemset node) will be created. These GI nodes will contain general itemsets that can be created from the half general itemsets by converting operators (=, <=) to operators (!=, >) respectively. Similar to the process of creating half general itemsets, we can generate general itemsets inside GI nodes in such a way that we can have random access to them. This process is done by considering operators (=, <=) as low rank and by assuming that items of itemsets are

MineHalfGeneralItemsets (Signature)
1)Sort Simple Itemsets of the Signature;
2) For each simple itemset
Create corresponding HGI node;
3) For each HGI node
Create the half general itemsets in order using the method of figure 2;
4) For each half general itemset
Calculate support using Lemmas 1, 2

Fig. 1. Procedure: Mining half general itemsets inside a signature

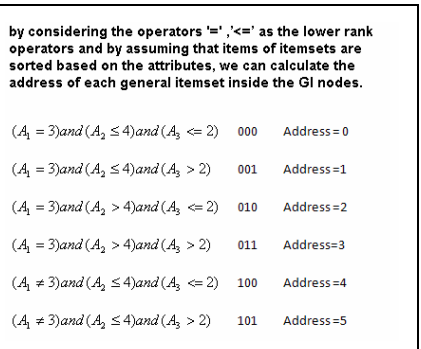


Fig. 2. Half general itemsets inside an HGI node

$(A_1 = 1) and (A_2 = 1)$ Support =10%
$(A_1 = 1) and (A_2 = 2)$ Support = 7%
$(A_1 = 1) and (A_2 = 3)$ Support =8%
$(A_1 = 2) and (A_2 = 1)$ Support =15%
$(A_1 = 2) and (A_2 = 2)$ Support = 31%
$(A_1 = 2) and (A_2 = 3)$ Support = 12%
$(A_1 = 3) and (A_2 = 1)$ Support = 7%
$(A_1 = 3) and (A_2 = 2)$ Support = 10%

Fig. 3. Simple itemsets of the Signature $\{A_1, A_2\}$

I_1 $(A_1 = 1) and (A_2 = 1)$ Support =10%
I_2 $(A_1 = 1) and (A_2 \le 1)$ Support =10%
I_3 $(A_1 \le 1) and (A_2 = 1)$ Support =10%
I_4 $(A_1 \le 1) and (A_2 \le 1)$ Support =10%

Fig. 4. Itemsets of the first HGI node

K_1 $(A_1 = 1) and (A_2 = 2)$ Support =7%
K_2 $(A_1 = 1) and (A_2 \le 2)$ Support =17%
K_3 $(A_1 \le 1) and (A_2 = 2)$ Support =7%
K_4 $(A_1 \le 1) and (A_2 \le 2)$ Support =17%

Fig. 5. Itemsets of the second HGI node

sorted based on the attribute ranking. Figure 7 illustrates the process of creating general itemsets inside a GI node. The corresponding half general itemset of the node in the figure is *'$(A_1=3)$ and $(A_2 \le 4)$ and $(A_3 \le 2)$'*. It is the first itemset of the GI node. As this itemset has operators $\{=, \le, \le\}$ (low rank operators), its address in the GI node will be 0. The next itemset will be generated from the above itemset by converting the operator of the last item from $\le$ to $>$. The created itemset has operators $\{=, \le, >\}$ for corresponding items (high rank operator for the last item), so its address in the GI node will be 001 or 1. The next itemset will have operators $\{=, >, \le\}$, so its address in the GI node will be 010 or 2 (high rank operator for the middle item). The fourth generated itemset is *'$(A_1=3)$ and $(A_2>4)$ and $(A_3>2)$'* which has operators $\{=, >, >\}$ and is located at address (011) (high rank operators for the last two items). The other itemsets will be created in the same manner.

After generating itemsets inside GI nodes, the next step is to calculate the supports of itemsets. The main difference between calculating supports of half general itemsets and general itemsets is the fact that each GI node must have addresses of its parents. The parents of a k-itemset are the k-1 subsets. For example, $(A_1=3)$ is a parent of the itemset $(A_1=3)$ and $(A_2=2)$. Having addresses of the parents facilitates the application of Lemmas 3 and 4. Figure 6 shows an algorithm to extract general itemsets.

MineGeneralItemset (Signature)
1) for each half general Itemset:
- Create Corresponding GI node
- Find the address of its parents

2) For each HGI node :
- For each GI node:
  - Create general itemsets according to figure 4;
  - Calculate the support of each general itemset using Lemmas 3 and 4
  -

Fig. 6. The procedure of mining general itemsets of a signature

by considering the operators '=' ,'<=' as the lower rank operators and by assuming that items of itemsets are sorted based on the attributes, we can calculate the address of each general itemset inside the GI nodes.

$(A_1 = 3) and (A_2 \le 4) and (A_3 <= 2)$	000	Address = 0
$(A_1 = 3) and (A_2 \le 4) and (A_3 > 2)$	001	Address =1
$(A_1 = 3) and (A_2 > 4) and (A_3 <= 2)$	010	Address =2
$(A_1 = 3) and (A_2 > 4) and (A_3 > 2)$	011	Address=3
$(A_1 \ne 3) and (A_2 \le 4) and (A_3 <= 2)$	100	Address =4
$(A_1 \ne 3) and (A_2 \le 4) and (A_3 > 2)$	101	Address =5

Fig. 7. General itemsets inside a GI node

5.3 Time Complexity and Memory Management of the MGR Algorithm

Mining half general itemsets from simple itemsets is approximately of linear complexity with regard to the number of simple itemsets. For each simple k_itemset, there are 2^k half general itemsets. According to Lemma 2, in order to compute the support of each half general itemset we need to search two half general itemsets. The complexity of searching one of them is log(s) where s is the average number of itemsets in signatures. The other itemset is located in the same HGI node as we have random access to half general itemsets of an HGI node, so its time complexity is negligible. Hence the overall time complexity of mining half general itemsets is equal to $2^k n(\log_2 s)$ where n is the number of simple itemsets and k is a constant with its maximum value equal to the number of attributes. Here $2^k n$ is the average number of half general itemsets.

The average number of half general itemsets in signatures can be calculated from the following relation:

s= (total number of half general itemsets)/ (number of signatures)

where (number of signatures) $= 2^{|I|} - 1$, (number of half general itemsets) $\le n.2^{|I|}$

To generate general itemsets, we should first sort the signatures ($(2^{|I|}-1)\log(2^{|I|}-1)$). Then according to the section 5.2, we should find all the k-1_itemsets for each half general k_itemset ($2^k n.k.\log_2 s$), finally the algorithm must calculate the supports of all half general itemsets ($4^k n 2^{k-1}(1+k)$).

So, the overall time complexity of the MGR algorithm is equal to

$$\Theta(n\log_2 s)+\Theta(n\log_2 n)+\Theta(n) \text{ or by substituting s,}$$
$$\Theta(n\log_2 n)+\Theta(n\log_2 n)+\Theta(n) \text{ or in fact } \Theta(n.\log n).$$

The other advantage of using an MGR tree is partitioning the problem into smaller ones. As seen in previous sections, mining itemsets in each signature can be done independently to the other signatures. It means that in each phase of mining itemsets, only holding one signature and its ancestors in main memory is sufficient. From the memory management point of view, it means that the MGR algorithm can be applied to large datasets without the need to hold all itemsets in main memory. From a parallel processing point of view, it means that mining itemsets in signatures can be done by different processors without any deadlock.

6 Experimental Results

6.1 Requirements of the Algorithm

The first step of the MGR algorithm, which is about mining simple itemsets, has great effect on the output of the algorithm. If there is no restriction on the supports of simple itemsets, the MGR algorithm can extract all general itemsets. If we set a non-zero value for support-threshold of simple itemsets, some infrequent itemsets will not be presented to the MGR algorithm. Absence of these itemsets has two effects on the output of the algorithm. Firstly, the algorithm ignores constructing HGI nodes corresponding to those simple itemsets which leads to the loss of all general itemsets belonging to those HGI nodes. Secondly, it causes errors in calculating the supports of general itemsets because the absent itemsets have an effect on supports of general itemsets. In order to achieve high quality rules, the support-threshold of simple itemsets must be low. In the experimental results of the following sections, the support threshold of simple itemsets is set to zero.

6.2 Experimental Results

In order to present the efficiency of the MGR algorithm, we compare it with an extension of the Apriori algorithm which mines general itemsets over combinations of base intervals similar to [11]. The Apriori algorithm is implemented using the Trie data structure and has better performance than many known algorithms [4]. It is designed so that its output is similar to the output of the MGR algorithm. This helps effective comparison of the algorithms. Extending the FP-growth algorithm [7] to mine general

itemsets has difficulties because each branch of the FP-tree will contain items with similar attributes. For example, a record that has value 1 for attribute "A_1" can cover items {(A_1=1), (A_1<=1), (A_1<=2), (A_1<=3), (A_1<=4), (A_1<>2), (A_1<>3),..} etc.

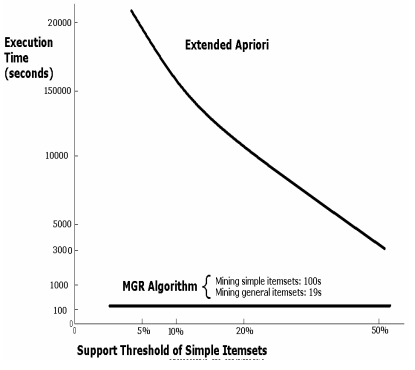


Fig. 8. MGR Vs Extended Apriori on synthetic dataset

Table 2. Properties of the synthetic dataset

Number of non class attributes	4
Domain values of non-class attributes	{0,1,2,3,4}
Number of classes	3
Number of records	100,000
Error ratio	0.1
Missing values ratio	0

All such items will exist in each branch of the FP-tree which causes trouble, as each itemset must have just one item with A_1 attribute. In order to avoid such difficulties, we do not use the FP-growth algorithm. We apply the algorithms on a synthetic dataset which is created using the DataGen tool [12]. Table 2 contains the details of the dataset. Figure 8 represents the execution time when applying both the extended Apriori and MGR algorithms to extract general itemsets from the synthetic dataset. The support threshold of simple itemsets using the MGR algorithm was set to zero.

As can be seen, the execution time of the MGR algorithm on the synthetic dataset is more than an order of magnitude lower than for extended Apriori. The total execution time of MGR is less than 120 seconds, while the execution time of extended Apriori with supports higher than 5% is about 21167 seconds. It can be inferred that the performance of the MGR algorithm is independent of the number of records or the size of the dataset (see figures 9 and 10). Figures 9 and 10 represent the execution time of the MGR algorithm with respect to the number of records. Figure 9 shows the total execution time of the MGR algorithm, which consists of mining simple itemsets and mining general itemsets. Figure 10 only shows the execution time for mining general itemsets from simple ones. As can be seen, the total execution time of the algorithm is almost linear irrespective of the number of records. It is easily inferred from figures 9 and 10 that mining general itemsets from simple itemsets is approximately constant and in fact, it is mining simple itemsets that is linear with regards to the number of records.

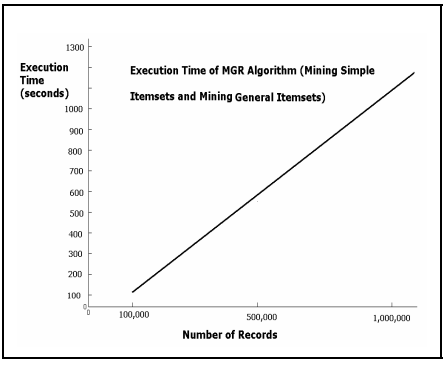


Fig. 9. Total MGR execution time

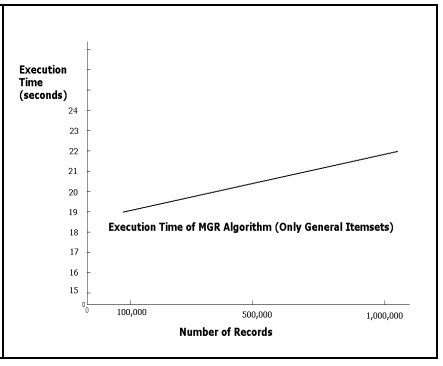


Fig. 10. Mining general items

7 Conclusions

In this paper, we proposed a time and space-efficient new algorithm for mining general association rules from tabular data. Decomposing the problem into several sub problems and employing the MGR tree makes the algorithm efficient in terms of time-complexity and memory requirements. The possibility of holding most of the MGR tree in secondary memory (hard disk) also makes the algorithm more space-efficient. In particular, it was shown that the algorithm stores candidate general itemsets in a tree structure in such a way that supports of complex itemsets can be recursively computed from supports of simpler itemsets.

As general rules can have equality and other comparison operators like $\{\leq, \geq, <, >, \neq, =\}$, we can discover more sophisticated patterns in data. As general rules have higher support and confidence than simple ones, they can represent more powerful patterns. In this paper, we have shown the power of general association rules to describe data, however we have not yet offered an approach to prune unnecessary general rules. More experiments will be done to compare the general and simple rules extracted from the Balance dataset [8]. In addition, further work will address ways for pruning insignificant rules as well as the potential for parallelizing the algorithm.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: SIGMOD Conference, pp. 207–216 (1993)
2. Richards, G., Rayward-Smith, V.J.: Discovery of Association Rules in Tabular Data. In: IEEE International Conference on Data Mining (2001)
3. Berzal, F., Cubero, J-C., Marin, N., Serrano, J.: TBAR: an Efficient Method for Association Rule Mining in Relational Databases. Data and Knowledge Engineering. 37, 47–64 (2001)

4. Bodon, F.: A Trie-based APRIORI Implementation for Mining Frequent Item Sequences. In: 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations, pp. 56–65 (2005)
5. Rastogi, R., Shim, K.: Mining Optimized Support Rules for Numeric Attributes. In: 15th International Conference on Data Engineering, pp. 126–135. IEEE Computer Society Press, Sydney, Australia (1999)
6. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: VLDB Conference, pp. 487–499 (1994)
7. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns without Candidate Generation. In: ACM SIGMOD Conference, pp. 1–12 (2000)
8. Merz, C.J., Murphy, P.: UCI Repository of Machine Learning Databases (1996), http://www.cs.uci.edu/mlearn/MLRepository.html
9. Hsu, P., Chen, Y., Ling, C-C.: Algorithms for Mining Association Rules in Bag Databases. Information Sciences 166(1-4), 31–47 (2004)
10. Aumann, Y., Lindell, Y.: A Statistical Theory for Quantitative Association Rules. Journal of Intelligent Information Systems 20(3), 255–283 (2003)
11. Sirkant, R., Agrawal, R.: Mining Quantitative Association Rules in Large Relational Tables. In: SIGMOD, pp. 1–12 (1996)
12. Melli, G.: DataGen Tool (2004), http://www.datasetgenerator.com/
13. Ke, J., Cheng, J., Ng, W.: Mining Quantitative Correlated Patterns using an Information Theoretic Approach. In: 12th ACM SIGKDD KDD Conference, Philadelphia, PA, USA, pp. 227–236 (2006)
14. Richards, G., Rayward-Smith, V.J.: The Discovery of Association Rules from Tabular Databases Comprising Nominal and Ordinal Attributes. Intelligent Data Analysis 9(3), 289–307 (2005)

Intrusion Detection at Packet Level by Unsupervised Architectures

Álvaro Herrero[1], Emilio Corchado[1], Paolo Gastaldo[2], Davide Leoncini[2], Francesco Picasso[2], and Rodolfo Zunino[2]

[1] Department of Civil Engineering, University of Burgos
C/ Francisco de Vitoria s/n, 09006 Burgos, Spain
{ahcosio,escorchado}@ubu.es
[2] Dept. of Biophysical and Electronic Engineering (DIBE), Genoa University
Via Opera Pia 11a, 16145 Genoa, Italy
{paolo.gastaldo,francesco.picasso,rodolfo.zunino}@unige.it

Abstract. Intrusion Detection Systems (IDS's) monitor the traffic in computer networks for detecting suspect activities. Connectionist techniques can support the development of IDS's by modeling 'normal' traffic. This paper presents the application of some unsupervised neural methods to a packet dataset for the first time. This work considers three unsupervised neural methods, namely, Vector Quantization (VQ), Self-Organizing Maps (SOM) and Auto-Associative Back-Propagation (AABP) networks. The former paradigm proves quite powerful in supporting the basic space-spanning mechanism to sift normal traffic from anomalous traffic. The SOM attains quite acceptable results in dealing with some anomalies while it fails in dealing with some others. The AABP model effectively drives a nonlinear compression paradigm and eventually yields a compact visualization of the network traffic progression.

Keywords: Intrusion Detection System, Network Security, Vector Quantization, Self-Organizing Map, Auto Associative Back Propagation.

1 Introduction

Automatic detection of anomalous traffic is one of the crucial topics in the area of network communication. Intrusion Detection Systems (IDS's) [1] are designed to monitor the traffic in computer networks and generate alerts, or trigger defensive actions, when suspect activities are detected. Nowadays, IDS's have become common elements in modern infrastructures to enforce network policies; nonetheless, some scientific issues remain open in IDS's development and run-time operation.

Today's IDS implementations address either misuse intrusion detection (MID) or anomaly intrusion detection (AID) [1]. MID systems recognize known attack patterns, and typically discriminate normal from malicious traffic by using a knowledge base of rules. MID suffers from two basic drawbacks: the set of rules is susceptible to inconsistencies and continuous updating is required to incorporate unseen attack patterns. From a different perspective, AID systems embed a model of 'normal'

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 718–727, 2007.

traffic and generate alerts when 'abnormal' events are detected. These techniques do not use sets of rules, and can support time-zero detection of novel attack strategies; however, AID systems require consistent modeling of normal traffic. Accuracy in detection proves indeed the major limitation of such approach [1]. To circumvent that issue, data-driven techniques have been applied to IDS's models; in particular, connectionist models (supervised and unsupervised approaches) have been profitably used. Supervised methods [1], [2], [3] tackle intrusion detection as a binary classification problem (i.e., normal vs. abnormal traffic). They can attain quite accurate results; in fact, the need for data labeling in the set-up phase and the continuous evolution of attack types often lead to a very expensive training process. Unsupervised methods [1], [4], [5] first extract features from traffic data and then apply unlabelled learning methods: the goal is to identify the significant portions of the feature space that support the distribution of normal traffic, whereas outliers will mark abnormal traffic activities. Unsurprisingly, supervised methods outperform unsupervised approaches at identifying known patterns [1]; by contrast, the latter ones prove more robust when coping with unknown attacks in a dynamic scenario, and therefore have been chosen as the scientific baseline for the present research.

This paper tackles the anomaly detection task by analyzing the performance of three different unsupervised paradigms: Vector Quantization (VQ) [6], Self-Organizing Maps (SOM) [7] and Auto-Associative Back-Propagation (AABP) neural networks [8]. These unsupervised paradigms have been previously applied to intrusion detection but the novelty of this paper is based on the issue that they deal for the first time with packet datasets. The VQ model represents a powerful technique to support the basic space-spanning mechanism, i.e. normal traffic vs. anomalous traffic. The SOM focus the intrusion detection task from a similar perspective as it generates a 2D mapping that preserves the topological properties of the input space. Finally, the AABP-based approach tackles the anomaly-detection problem in terms of dimensionality reduction, thus supporting a nonlinear compression that eventually leads to a compact visualization of the network traffic evolution.

The experimental domain involves both normal traffic and anomalous traffic ascribed to the Simple Network Management Protocol (SNMP), which represents one of the top 5 most vulnerable services [9]. Empirical tests involved a dataset previously used in literature for unsupervised analysis [10], [11]. As it has been previously mentioned, a great amount of connectionist models (including VQ, SOM and AABP) have been already applied to intrusion detection. They have been applied to the KDD dataset [12], which contains information about TCP connections. On the contrary, in this work, unsupervised learning is applied to a dataset containing information from the packet level. That is, information extracted from the header of network packets, providing a complementary intrusion detection point of view.

2 A Connectionist-Based Framework for IDS

The general scheme for the proposed connectionist-based framework for intrusion detection can be summarized as follows (Fig. 1):

- packets traveling through the network are intercepted by a capture device;
- traffic is coded by a set of features spanning a multidimensional vector space;
- the connectionist model operates on feature vectors and yields as output a suitable representation of the network traffic.

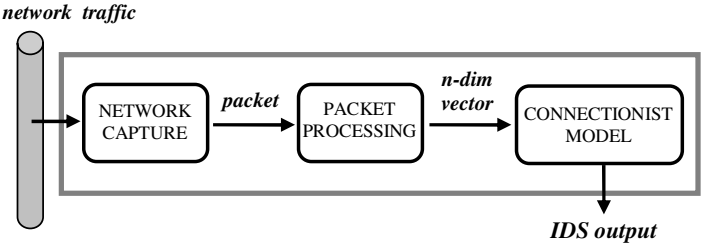


Fig. 1. The connectionist-based IDS framework

The connectionist model clearly is the actual core of the overall IDS. That module is designed to yield an effective representation of network traffic, thus providing a powerful tool for the automated identification of traffic anomalies. Hence, the effectiveness of the overall approach strictly relates to the successful support to the network supervisor at detecting offending attacks. In the proposed scheme, the connectionist component processes an *n*-dimensional vector that has been previously assembled by a "packet processing" module, which extracts numerical features associated with each network packet. Thus, as a novelty, the proposed IDS operates at the packet level and not at the connection level as other models do [12], [13], [14]. The design of the feature set is indeed a crucial issue [15]. In principle timestamp, source and address port, and protocol can uniquely identify a connection [16]. When dealing with Transmission Control Protocol (TCP) traffic, additional features may be required (e.g. to track connection state [17]); instead, User Datagram Protocol (UDP) traffic can be effectively characterized by a reduced feature set [10].

A connectionist approach appears consistent with the AID problem setting mainly because it allows a system to empirically learn the input-output relationship between raw traffic and subsequent interpretation. The crucial advantage is that the eventual outlier-detection method does not require any *a-priori* analytical formulation of the underlying phenomenon. In principle, any unsupervised method applies to the involved representation process; actually, this work explores the performance of three specific paradigms: Vector Quantization, Self-Organizing Maps and Auto-Associative Back-Propagation.

3 Unsupervised Methods for IDS Implementation

3.1 Vector Quantization

VQ is important in high-dimensional information processing, since represented data and representation codes are expressed in the same domain. The information-representation paradigm aims to partition the data space into several portions, each

one identified by a specific reference codevector, or prototype; hence, VQ is very useful for data representation and compression applications.

The crucial point determining a VQ system performance is represented by the codevector-positioning algorithm. The classical formulation of the VQ training problem can be set as follows. The n-dimensional data space is partitioned by a set of prototypes, $V=\{\mathbf{v}_i \in \Re^n,\ i=1,..,N_p\}$, which lie at "significant" positions in the data space; each prototype covers the samples lying within its associate partition. To assign a prototype to each sample, a best-match criterion minimizing a distortion cost is used. Euclidean metrics is usually adopted to measure distortion; hence a data sample, $\mathbf{x} \in \Re^n$, is associated with the prototype, $\mathbf{v}^*(\mathbf{x}) \in V$, that satisfies

$$\mathbf{v}^*(\mathbf{x}) = \arg\min_{\mathbf{v} \in V} \left\{ \|\mathbf{x} - \mathbf{v}\|^2 \right\} \tag{1}$$

The VQ-representation problem implies finding the optimal codebook, V^+, that minimizes the overall distortion:

$$\min E(P) = \int_{\Re^n} \left\| \mathbf{x} - \mathbf{v}^*(\mathbf{x}) \right\|^2 p(\mathbf{x}) d\mathbf{x} \tag{2}$$

The actual sample distribution $p(\mathbf{x})$ is not known *a priori*, hence the integral cannot be computed analytically in any but very peculiar cases. Therefore, one usually resorts to an empirical estimation of the involved distortion: a set of training samples, $X=\{\mathbf{x}_l \in \Re^n,\ l=1,..,N_d\}$, drives vector positioning to minimize the empirical cost:

$$\min\ \hat{E}(V) = \frac{1}{N_p} \sum_{i=1}^{N_p} \left\| \mathbf{x}_i - \mathbf{v}^*(\mathbf{x}_i) \right\|^2 \tag{3}$$

In this research, the VQ training algorithm is based on the "Plastic Neural Gas" (PGAS) [6] model for neural network training. The PGAS algorithm is an adaptive version of the conventional 'neural gas' framework [18], to which it added the ability of dynamically creating and deleting prototypes. The PGAS method is guaranteed to converge in a finite number of steps and, as compared with the neural gas approach, does not suffer from the problem of "dead vectors."

3.2 Self-organizing Maps

The well known Self-Organizing Map (SOM) [7] is composed of a discrete array of L nodes arranged on an N-dimensional lattice and it maps these nodes into D-dimensional data space while maintaining their ordering. The dimensionality, N, of the lattice is normally smaller than that of the input data. Thus, the SOM provides low dimensional representations of multi-dimensional datasets while preserving the topological properties of the input space. The SOM is based on a type of unsupervised learning called competitive learning; an adaptive process in which the neurons in a neural network gradually become sensitive to different input categories, sets of samples in a specific domain of the input space [19]. This can be seen in the SOM update of neighbourhood neurons:

$$w_k(t+1) = w_k(t) + \alpha(t)\eta(v,k,t)\big(x(t) - w_v(t)\big) \tag{4}$$

where w_v is the winning neuron, α the learning rate of the algorithm, $\eta(v,k,t)$ is the neighbourhood function where v represents the position of the winning neuron in the lattice and k the positions of the neurons in the neighbourhood of this one, x is the input to the network,

To evaluate the adaptation quality of the map to the dataset, two different measures have been used: topographic [20] and quantization [21] error.

3.3 Auto-associative Back-Propagation Networks

Auto-Associative Back-Propagation (AABP) networks constitute an unsupervised variant of the general Multi-Layer Perceptron (MLP) model, which belongs to the feedforward class of neural networks [23]. A conventional MLP includes three layers (input, 'hidden' and output), and associates an input vector, $\boldsymbol{x} \in \Re^D$, with an output vector, $\boldsymbol{y} \in \Re^Q$, computed as:

$$y_q(\mathbf{x}) = w'_{q,0} + \sum_{u=1}^{N_h}\left[w'_{u,q} \cdot \sigma\left(w_{u,0} + \sum_{k=1}^{D} w_{u,k} x_k \right)\right]; q = 1,...,Q \tag{5}$$

where $\sigma()$ is the sigmoidal function, N_h is the depth of the sigmoid series expansion, and W represents the coefficients of the weights for the interconnections between the layers. The empirical Back-Propagation (BP) algorithm [23] drives the weights, W, so that the network best reproduces the desired mapping over a given training set.

The AABP model forces target outputs to coincide with the network inputs: $\mathbf{t} \equiv \mathbf{x}$., hence the network should replicate the training sample distribution Since the hidden layer is typically smaller than the input/output ones, the goal is to reduce data dimensionality. At run-time, an AABP network associates each input vector with the 'coding' values computed by the hidden neurons, and therefore supports a (lossy) compression of input data into a lower-dimensional space. The sophisticated AABP model proposed in [8] involves a Non-Linear Principal Component Analysis (NLPCA) architecture.

The crucial difference from classical AABP is that the mapping and reconstruction sections include an additional layer of neurons. The NLPCA architecture retains the universal approximation ability of BP networks [8]. At run-time, the five-layer resulting network operates in the same way as a three-layer AABP: the outputs of the 'coding' layer yield the low-dimensional representation of each input vector.

The increased power of representation conveyed by the NLPCA augmentation is remarkable. The main advantage is that the compressed representation does not relate to any linear model (as Principal Component Analysis [22]). Non-Linear techniques fit those domains in which a non-linear representation best encompasses the observed empirical phenomenon.

4 Unsupervised Connectionist Methods for IDS's

The main demonstration domain of the present research involves traffic anomalies within SNMP, which is a part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite. SNMP supports the exchange of management information

between network devices at the application layer. SNMP data are liable to attacks that may compromise system security [9]. This paper addresses two types of attacks relying on the SNMP protocol:

- SNMP port scan/sweep: a port scan attempts to count the services running on a group of machines by probing ports for response. A port sweep provides information on security weaknesses.
- MIB information transfer: the Management Information Base (MIB) is a collection of information concerning managed devices, including sensitive data. SNMP is used to access MIB objects.

4.1 The Feature Set

The eventual network-based IDS for the detection of SNMP anomalous traffic is structured as shown in Figure 1. The "feature extraction" component (Section 2) generates feature vectors by working out information contained in the packet header. In the present research, network packets are characterized by using the set of features that already proved to be effective for detection of anomalous SNMP traffic [10]. These features can be listed as follows:

- Protocol ID: an integer number that identifies the protocol of the packet.
- Source port: the port number of the device that sent the packet.
- Destination port: the port number of the host to which the packet is sent.
- Size: the packet size (in Bytes).

The four-dimensional space feeds the connectionist component, which is entitled to embed the intrusion-detection task. In the following, the set-up of the VQ-based system and the AABP-based system are introduced.

4.2 Set-Up of the Unsupervised Connectionist Models

The VQ paradigm represents a powerful tool to tackle the crucial task that characterizes an AID model: the definition of the 'normal' traffic profile. Thus, the VQ-based IDS is designed to operate as a 'smart' traffic analyzer, which automatically identifies anomalous traffic, i.e. network packets that do not belong to the normal profile. The training phase uses empirical data to configure the prototype set V^+, which should single out the normal profile in the 4-dimensional space that characterizes the network packets. At run-time, the VQ-based component classifies the incoming network packets according to the normal profile defined in the training phase; thus, apposite alarms are generated when a network packet lies in a sector of the 4-dim space that is not associated to the normal profile.

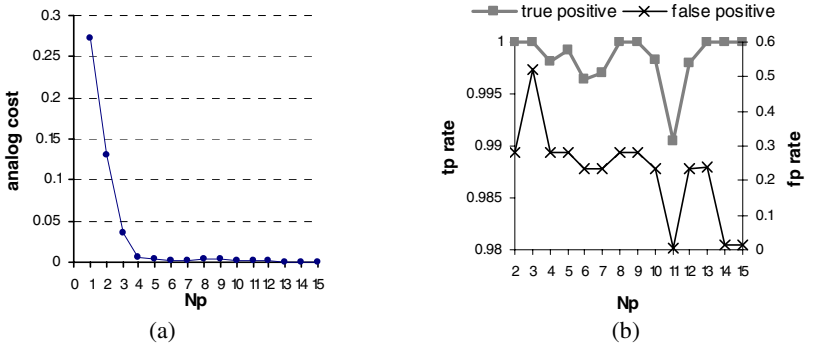
As compared with the VQ paradigm, AABP neural networks represent an intriguing alternative for unsupervised learning, especially when considering its non-linear formulation [8]. The present research exploits the NLPCA architecture to generate a two-dimensional representation of the network traffic by starting from the four-dimensional space defined by the feature set. Hence, the AABP neural network supports the mapping of raw data extracted from traffic sources into an intuitive visual representation.

5 Experimental Results

The three unsupervised approaches to IDS development were tested by using the data set used in [10]. This data set contained network packets captured from UDP traffic, as SNMP uses UDP as the transport protocol for passing data between managers and agents. Hence, the data set included only packets using UDP as transport layer and IP as network layer, and a total of 5866 samples (i.e. network packets) spanned a four-dimensional feature space.

5.1 VQ Paradigm

The PGAS training algorithm drove the unsupervised partitioning of the four-dimensional space. The present research exploited the plasticity feature of PGAS to properly size the cardinality of the prototype set; to this purpose, Figure 2.a reports the results obtained by the training phase. The graph gives on the *x*-axis the cardinality, N_p, of the prototype set, which dynamically grows; the corresponding analog cost is plotted on the *y*-axis. Figure 2.a shows that the analog cost dramatically decreases in the range $1 \leq N_p \leq 4$; then, for $N_p > 4$ the curve is characterized by an asymptotic behavior. $N_p = 14$ appears to be the suitable cardinality of the eventual prototype set V^+, as the curve exhibits small oscillations for $6 \leq N_p \leq 12$.



(a) (b)

Fig. 2. Results obtained by the PGAS algorithm: (a) training phase; (b) test results

To provide a qualitative assessment, Figure 2.b presents the results obtained on the test set by displaying the true-positive (tp) rate and the false-positive (fp) rate for the different cardinalities of the prototype set. The graph clearly shows that the best performance (tp = 100% and fp = 0.01%) is attained with $N_p \geq 14$. Thus, the PGAS algorithm proved to be a powerful tool to tackle anomalous traffic detection.

5.2 Self-organizing Map

The SOM was applied to the previously described data set. A 30x20 neuron lattice was generated to perform the dimensionality reduction. The mapping obtained after training is shown in Figure 3. For a better understanding of results, the neurons have

been labeled (See Fig. 3.a) with class information. Each instance in the dataset includes the class value according to: C1 - normal traffic, C2 - port sweeps and C3 - MIB transfer.

As can be seen in Figure 3.a, there are 3 groups of neurons (Groups 1, 2 and 3) modeling most of the normal traffic (C1). All the packets related to the port sweeps (C2) are identified by neurons constituting Groups 4 and 5. The rest of the neurons identify packets related to normal traffic as well as packets related to the MIB information transfer (C3). Figure 3.b shows the associated U-matrix.

The quality measures associated to the results shown in Figure 3 are: quantization error= 0.011 and topographic error= 0.2.

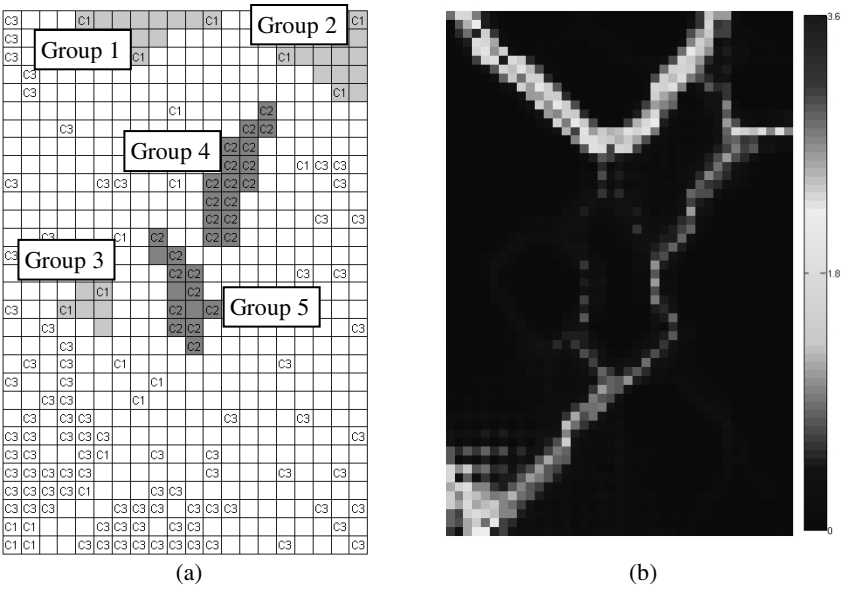


Fig. 3. Best results obtained by the SOM: (a) labeled map and (b) associated U-matrix

5.3 AABP Paradigm

According to the set-up introduced in Section 4, the eventual AABP-based IDS was trained to map the original four-dimensional space into a two-dimensional space for an intuitive visualization of the traffic progress. In the experiments presented here, the configuration of the AABP network included a number of 30 nodes in the hidden layers (coding and reconstruction), while of course the number of neurons in the middle layer was N_h=2. Although theoretical studies did not succeed in providing any established design criterion to set the number of a network's hidden nodes, the literature provides practical criteria [24] for dimensioning a network size, in order to ensure prediction accuracy while minimizing the risk of overfitting training data. In summary, the architecture of the overall AABP network was set as follows: 4 nodes in the input layer, 30 nodes in the compression layer, 2 nodes in the coding layer, 30 nodes in the decompression layer, and 4 nodes in the output layer.

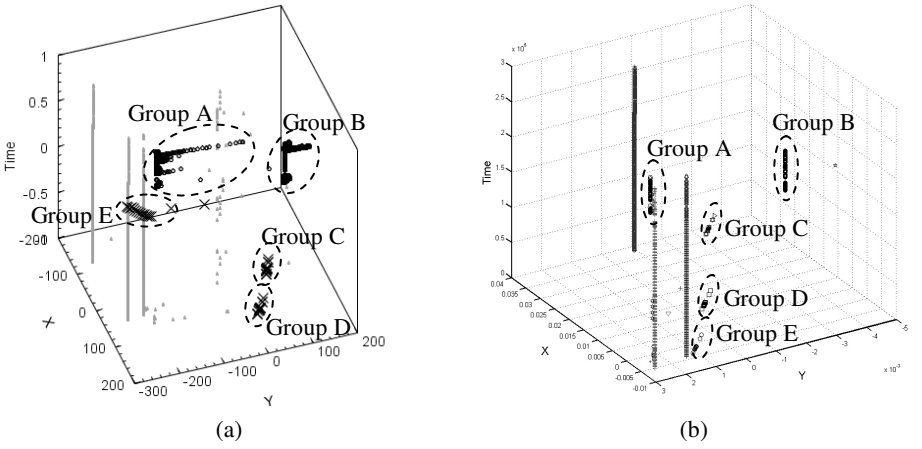


Fig. 4. Traffic visualization: (a) NLPCA projection; (b) PCA projection

Figure 4 shows the projection obtained by NLPCA (Fig. 4.a) and PCA (Fig. 4.b). The graph gives on the x-axis and y-axis the outputs of the two neurons in the middle layer, i.e. the compressed representation of the signal; besides, the z-axis gives the time evolution of the packets. In Figure 4.a, gray markers characterize the normal traffic, while black crosses are used to mark port sweeps and black circles are used to mark the MIB information transfer. One can easily note that the proposed NLPCA-based system manages to identify the two anomalous situations contained in the data set: the MIB information transfer (Groups A and B in Figure 4.a) is identified due to its orthogonal direction with respect to the normal traffic and to the high density of packets; the sweeps (Groups C, D and E in Figure 4.a) are identified due to their non-parallel direction to the normal one. In this type of visualization, normal traffic is associated to straight lines evolving in parallel directions.

For comparison purposes, PCA was also applied to the same problem. As can be seen in Figure 4.b, PCA was only able to identify the port sweeps (Groups C, D and E), while it failed to detect the MIB information transfer (Groups A and B) because the packets in this anomalous situation evolve in the same way (a parallel direction) as the "normal" one.

Acknowledgments. This research has been partially supported by the MCyT project TIN2004-07033.

References

1. Laskov, P., Dussel, P., Schafer, C., Rieck, K.: Learning Intrusion Detection: Supervised or Unsupervised? In: Roli, F., Vitulano, S. (eds.) ICIAP 2005. LNCS, vol. 3617, pp. 50–57. Springer, Heidelberg (2005)
2. Liao, Y., Vemuri, V.R.: Use of K-nearest Neighbor Classifier for Intrusion Detection. Comput. Security 21(5), 439–448 (2002)

3. Sarasamma, S.T., Qiuming, A.Z., Huff, J.: Hierarchical Kohonen Net for Anomaly Detection in Network Security. IEEE Trans. on SMC – part B 35(2) (2005)
4. Zanero, S.: Analyzing TCP Traffic Patterns Using Self Organizing Maps. In: Roli, F., Vitulano, S. (eds.) ICIAP 2005. LNCS, vol. 3617, pp. 83–90. Springer, Heidelberg (2005)
5. Zheng, J., Hu, M.: An Anomaly Intrusion Detection System Based on Vector Quantization. ICIE Trans. on Inf. & Syst. E89-D(1) (2006)
6. Ridella, S., Rovetta, S., Zunino, R.: Plastic Algorithm for Adaptive Vector Quantization. Neural Computing & Applications 7, 37–51 (1998)
7. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE 78(9), 1464–1480 (1990)
8. Kramer, M.A.: Nonlinear Principal Component Analysis using Autoassociative Neural Networks. AIChE Journal 37(2) (1991)
9. Cisco Secure Consulting: Vulnerability Statistics Report (2000)
10. Corchado, E., Herrero, A., Saiz, J.M.: Detecting Compounded Anomalous SNMP Situations using Unsupervised Pattern Recognition. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) ICANN 2005. LNCS, vol. 3697, pp. 905–910. Springer, Heidelberg (2005)
11. Corchado, E., Herrero, A., Saiz, J.M.: Testing CAB-IDS through Mutations: on the Identification of Network Scans. In: Gabrys, B., Howlett, R.J., Jain, L.C. (eds.) KES 2006. LNCS (LNAI), vol. 4252, pp. 433–441. Springer, Heidelberg (2006)
12. Elkan, M.: Results of the KDD 1999 Classifier Learning Contest (1999), online from: http://www-cse.ucsd.edu/users/elkan/clresults.html
13. Lippmann, R., Haines, J.W., Fried, D.J., Korba, J., Das, K.: Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation. In: Debar, H., Mé, L., Wu, S.F. (eds.) RAID 2000. LNCS, vol. 1907, pp. 162–182. Springer, Heidelberg (2000)
14. Sabhnani, M., Serpen, G.: Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context. In: Proc. MLMTA 2003, pp. 623–630 (2003)
15. Lee, W., Xiang, D.: Information-Theoretic Measures for Anomaly Detection. In: Proc. 2001 IEEE Symp. on Security and Privacy, pp. 130–143 (2001)
16. Lee, W., Stolfo, S.J., Mok, K.W.: Mining in a Data-Flow Environment: Experience in Network Intrusion Detection. In: KDD 1999. Proc. 5th ACM International Conference on Knowledge Discovery and Data Mining, pp. 114–124 (1999)
17. Lee, W., Stolfo, S.J., Mok, K.W.: Adaptive Intrusion Detection: A Data Mining Approach. Artificial Intelligence Review 14(6), 533–567 (2000)
18. Martinetz, T., Berkovich, S.G., Schulten, K.J.: Neural Gas Network for Vector Quantization and its Application to Time-Series Prediction. IEEE TNN 4(4), 558–569 (1993)
19. Kohonen, T., Lehtio, P., Rovamo, J., Hyvarinen, J., Bry, K., Vainio, L.: Principle of Neural Associative Memory. Neuroscience 2(6), 1065–1076 (1977)
20. Kiviluoto, K.: Topology Preservation in Self-Organizing Maps. In: IEEE International Conference on Neural Networks, vol. 1, pp. 294–299 (1996)
21. Kohonen, T.: Self-Organizing Maps. Springer Series In Information Sciences, vol. 30. Springer, New York (1997)
22. Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine 2(6), 559–572 (1901)
23. Rumelhart, D.E., McClelland, J.L.: Parallel Distributed Processing. MIT Press, Cambridge, MA (1986)
24. Widrow, W., Lehr, M.A.: 30 Years of Adaptive Neural Networks: Perceptron, Madaline and Back Propagation. Proc. IEEE 78(9), 1415–1442 (1990)

Quality of Adaptation of Fusion ViSOM

Bruno Baruque[1], Emilio Corchado[1], and Hujun Yin[2]

[1] Department of Civil Engineering. University of Burgos, Spain
bbaruque@ubu.es, escorchado@ubu.es
[2] School of Electrical and Electronic Engineering. University of Manchester, UK
h.yin@manchester.ac.uk

Abstract. This work presents a research on the performance capabilities of an extension of the ViSOM (Visualization Induced SOM) algorithm by the use of the ensemble meta-algorithm and a later fusion process. This main fusion process has two different variants, considering two different criteria for the similarity of nodes. These criteria are Euclidean distance and similarity on Voronoi polygons. The capabilities, strengths and weakness of the different variants of the model are discussed and compared more deeply in the present work. The details of several experiments performed over different datasets applying the variants of the fusion to the ViSOM algorithm along with same variants of fusion with the SOM are included for this purpose.

1 Introduction

A general way of boosting the stability and classification capabilities of classic classifiers (such as decision trees) is the construction of ensembles of classifiers [4], [5]. Following the idea of a 'committee of experts', the ensemble technique consists of training several identical classifiers on slightly different datasets in order to constitute a 'committee' to classify new instances of data.

Topology Preserving Maps [1], which include the Self-Organizing Maps (SOM) [2] and the Visualization Induced SOM (ViSOM) [3], were originally created as a visualization tool; enabling the representation of high-dimensional datasets onto two-dimensional maps and facilitating the human expert the interpretation of data. The main problem of these unsupervised techniques is their inherent instability. Even running the same algorithm using the same parameters and dataset can yield quite dissimilar results.

The ensemble meta-algorithm approach can be used on several topology preserving models to improve their stability and visualization performance. This is done by the training of several complementary networks and computing a fusion of them that outperforms each of its composing individually. This work main objective of this research is to present a study of the characteristics and performance of two different variants of the fusion process.

The rest of the paper is organized as follows: In section 2 the basics of the Self-Organizing Maps and its extension the Visualization Induced SOM, along with some quality measures for this kind of maps are presented. Section 3 is dedicated to the explanation of the ensemble training and the latter fusion process (which includes two

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 728–738, 2007.

variants). Section 4 includes the details of several experiments performed using several real datasets over the two different variants, with comparison of the strengths and weakness of each one. Finally, section 5 contains the conclusions and directions of future work extracted form the present work.

2 Quality Measures for Topology Preserving Models

2.1 The ViSOM Learning Algorithm

In this study, two different models will be applied: the Self-Organizing Map (SOM) and its variant the Visualization Induced SOM (ViSOM). Both the SOM and the ViSOM models belong to a family of techniques with a common target: to produce a low dimensional representation of the training samples while preserving the topological properties of the input space. The best known technique is the Self-Organizing Map algorithm [2]. It is based on a type of unsupervised learning called competitive learning; an adaptive process in which the neurons in a neural network gradually become sensitive to different input categories, sets of samples in a specific domain of the input space [1].

An interesting extension of this algorithm is the Visualization Induced SOM [3], [6] proposed to directly preserve the local distance information on the map, along with the topology. The ViSOM constrains the lateral contraction forces between neurons and hence regularises the interneuron distances so that distances between neurons in the data space are in proportion to those in the input space.

The difference between the SOM and the ViSOM hence lies in the update of the weights of the neighbours of the winner neuron as can be seen from Eq. (1) and Eq. (2). Update of neighbourhood neurons in SOM:

$$w_k(t+1) = w_k(t) + \alpha(t)\eta(v,k,t)\big(x(t) - w_v(t)\big) \tag{1}$$

Update of neighbourhood neurons in ViSOM:

$$w_k(t+1) = w_k(t) + \alpha(t)\eta(v,k,t)\left([x(t) - w_v(t)] + [w_v(t) - w_k(t)]\left(\frac{d_{vk}}{\Delta_{vk}\lambda} - 1\right)\right) \tag{2}$$

where w_v is the winning neuron, α the learning rate of the algorithm, $\eta(v,k,t)$ is the neighbourhood function where v represents the position of the winning neuron in the lattice and k the positions of the neurons in the neighbourhood of this one, x is the input to the network and λ is a "resolution" parameter, d_{vk} and Δ_{vk} are the distances between the neurons in the data space and in the map space respectively.

2.2 Quality Measures

To evaluate the quality of the adaptation of the map to the dataset represented several different measures have been devised [7]. A very widely known measure, used to indicate how good the units (neurons) of the map approximate the data on the dataset, is the mean square quantization error (MSQE). It is widely used as a measure of the

quality of vector quantization algorithms, but it can be easily adopted for SOM and ViSOM algorithms as represented in Eq. 3:

$$MSQE = \frac{1}{|D|} \sum_{x_i \in D} \left\| x_i - m_{b(x_i)} \right\|^2 \tag{3}$$

where $|D|$ is the number of data in the dataset D, and $m_{b(x_i)}$ is the best matching unit of the map to the data sample x_i of the dataset.

The other main characteristic of the Self-Organizing Maps is their topology preservation. As explained in [8], when using a constant radius for the neighbourhood function of the learning phase of a SOM; there exists a function that the algorithm optimizes. This function, called distortion measure in this work, can be used to measure the overall topology preservation of a map. It is computed as shown in Eq. 4,

$$E_d = \sum_{i=1}^{n} \sum_{j=1}^{m} h_{b_i j} \left\| x_i - m_j \right\|^2 \tag{4}$$

where $h_{b_i j}$ represents the neighbourhood function between the best matching unit and every other unit in the map.

Also the topology error measure will be mentioned in this work. It is one of the first and simplest topology measures. It consists on finding the first two best matching units for each entry of the dataset and testing whether the second is in the neighbourhood of the first or not. This can be computed as a normalized single value, indicating the overall quality of the map or decomposed to be visualized over each neuron of the map [7].

3 Topology Preserving Mapping Fusion

3.1 Use of the Ensemble Meta-algorithm

This technique was in its conception aimed at improving the performance of classification algorithms. It has been observed in several studies that although one of the classifiers in an ensemble would yield the best performance, the sets of patterns misclassified by the different classifiers would not necessarily overlap. This suggests that different classifier designs potentially offer complementary information about the patterns to be classified and could be harnessed to improve the performance of the selected classifier [4]. The main problem of competitive learning based networks is that are inherently unstable due to the nature of their learning algorithm. The leading idea of this work is that the effect of this instability may, however, be minimized by the use of ensembles [9]. The learning algorithm of the topology preserving maps family specifies that their composing units (or neurons) specialize during the algorithm iterations in recognizing a certain type of patterns, which determines also the topology of the map. Similarly to classification process, we can infer that the regions of the maps that do not represent trustfully the real nature of the dataset do not necessarily overlap. Therefore, the visualization of a single map might be improved by adapting each of the composing units of a map in the best way possible to the

dataset under study by using ensemble techniques, as they offer complementary visualizations of the maps.

Among the ensemble algorithms, the most complex types try to combine not only the results but the whole set of classifiers in order to construct a single better one that can outperform its individual components. In the case of this paper this perspective, the concept of a single "summary" or "synthesis" of the patterns stored within the whole ensemble, is the one followed to improve the model performance. The main objective is to obtain a unique map that may be seen to represent in the clearest and most reliable way as possible the different features contained in the different maps in the ensemble.

3.2 Fusion Variants Under Study

Several of ensemble techniques have been applied to the SOM [10], ViSOM [11] and other topological mapping networks, mainly for classification purposes.

Under the context of the visualization however, some adaptations are necessary to build a meaningful combination of the maps they represent. In this work a main algorithm for mapping fusion with two different variants is used for the first time in combination with the ViSOM. The objective is the comparison of the two in order to obtain conclusions that can be used in further studies to generate a more accurate model.

The procedure is the same for the training of the networks that compose the ensembles. All are trained using typical cross-validation, with the dataset divided into several folders, leaving one of them out to test the classification accuracy. To train the network ensemble the meta-algorithms called bagging [12] is used. It consists on obtaining n subsets of the training dataset through re-sampling with replacement and trains individual classifiers on such re-sampled subsets. This permits to generate n different trained networks which are combined into a final network that is expected to outperform each of them individually. The combination of maps is done once all the networks composing the ensemble have finished their training.

This combination is done in a neuron by neuron basis. That is, neurons that are considered 'near enough' one to the other are fused to obtain a neuron in the final fused network. This is done by calculating the centroid of the weights of the neurons to fuse:

$$w_{neuAvg} = 1/n \cdot \sum_{i=1}^{n} w(i) \tag{5}$$

That process is repeated until all neurons in all trained networks are fused into a unique final one. The criteria to determine which neurons are 'near enough' to be fused is what determines the two variants of the main algorithm.

Criterion 1: *Voronoi Polygons***.** Each neuron in a Self-Organizing Map can be associated with a portion of the input data space called the Voronoi polygon. That portion of the input multi-dimensional space is the portion that contains data for which that precise neuron is the Best Matching Unit (BMU) of the whole network [1]. It is therefore a logical conclusion to consider that neurons that are related to similar Voronoi polygons can be considered similar between them, as they should be situated relatively close in the input data space.

To calculate the dissimilarity between two neurons, a record of which data entries activated each neuron as the BMU can be kept. This can be done easily associating a

binary vector to the neuron which length is the size of the dataset and contains ones in the positions where the neuron was the BMU for that sample and zeros in the rest of positions. The dissimilarity (i.e. the distance) between neurons can therefore be calculated as:

$$d(v_r, v_q) = \frac{\sum_{l=1}^{n} XOR(v_r(l), v_q(l))}{\sum_{j=1}^{n} OR(v_r(j), v_q(j))} \tag{6}$$

being r and q the neurons to determine their dissimilarity and v_r and v_q the binary vectors relating each of the neurons with the samples recognized by it. A much more detailed explanation can be found in [13].

The main problem with this proximity criterion is that it depends on the recognition of data by the network, rather than on the network definition itself. This means that a neuron that does not react as the BMU for any data could be considered similar to another neuron with the same characteristic, although they can be relatively far from each other in the input data space. To avoid this, all neurons with a reacting rate lower than a threshold are removed before calculating the similarities between them. This implies that the neighbouring properties of the whole network are no longer considered. To keep a notion of neighbouring between the neurons of the fused network the similarity criteria must be used again. Neurons with dissimilarity less than a threshold will be considered as neighbours in the fused network.

Algorithm 1. *Fusion based on Similarity of the Voronoi Polygons*

1: Being nNet the number of networks to be fused
   fus the resultant fused network
   θu, θ_f and θ_c the usage, fusion and connection thresholds respectively
2: **for i=1:*nNet***
3: remove from the network the neurons that have a recognition rate lower than a usage threshold ($\Sigma_i v_r(i) < \theta_u$)
4: add all the rest of the neurons of network(i) to the set of all nodes of ensemble
5: **end**
6: calculate the dissimilarity between all neurons contained in the set obtained in 3-6 using Eq. 5
7: group in different sub-sets the nodes that satisfy that the dissimilarity between all of them is lower than the dissimilarity threshold and the distance between each of them and the rest of nodes in other sub-set is higher than that threshold.

$$\begin{cases} ds(v(n_r), v(n_q)) < \theta_f & \text{for all } n_r, n_q \in S_k \\ ds(v(n_r), v(n_q)) \geq \theta_f & \text{for all } n_r \in S_k,\ n_q \in S_l,\ k \neq l \end{cases}$$

   The result will be a set of sub-sets (*S*).
8: 'fuse' all the nodes in each sub-set to form a node of the final fused network by calculating the centroid of the nodes in each sub-set (see Eq. 4). The fused network will have as many nodes as sub-sets are contained in *S*.
9: create the final network (*fus*) including in it all the fused nodes
10: create connections between fused nodes in the fused network (*fus*) to represent neuron neighbourhood. Connections will be established if the distance between fused nodes is lower than the connection threshold, considering this distance as:

$$\min_{n_r \in N_r, n_q \in N_r} ds(v(n_r), v(n_q)) < \theta_c$$

Criterion 2: *Euclidean Distance.* This method involves comparing the networks neuron by neuron in the input space. This implies that all the networks in the ensemble must have the same size. First, it searches for the neurons that are closer in the input space (selecting only one neuron in each network of the ensemble) then it "fuses" them to obtain the final neuron in the "fused" map. This process is repeated until all the neurons have been fused. To deal with the high computational complexity of the algorithm, it can be implemented using dynamic programming. A more detailed description of this procedure can be found in [14].

The difference with the previous criteria is that, in this case, a pair wise match of the neurons of each network is always possible, so the final fused network has the same size as the single composing ones. This implies that a certain global neighbouring structure can be kept and reconstructed in the fused network.

Algorithm 2. *Fusion based on Euclidean Distance*

0: Train several networks by using the bagging (re-sampling with replacement) meta-algorithm
1: Being *nNet* the number of networks to be fused
   nNeur the number of neurons composing each network
   fus the resultant fused network
2: Initialize *fus* with the neuron weights of the first network
3: **for i=2:*nNet***
4: **for j=1:*nNeur***
5: *neuFus* : neuron (*j*) of the *fus* network
6: calculate Euclidean Distance (ED) between *neuFus* and ALL neurons of network(*i*)
7: *neuNet:* neuron with the minimum ED
8: calculate *neuAvg:* neuron whose weights are the average of the weights of *neuFus* and *neuNet* i.e. the centroid of both neurons' weights (see Eq. 4).
9: remove *neuNet* from the set of neurons of the network
10: replace *neuFus* by *neuAvg* in the *fus* network (in position *j* of the network)
11: **end for**
12: **end for**

---

4 Performance Experiments

To test the characteristics and capabilities of the fusion of ViSOM and compare both of its variants several real datasets have been employed. Data is extracted form the UCI repository [15] and include the Iris, the Wisconsin Breast Cancer and the Wine datasets.

Figures 1 to 3 depict a single ViSOM and the two variants of fusion represented over the iris dataset. Figure 4 represents the dataset plotted in 2 dimensions over the map obtained of 'unfolding' the network in Figure 2.

It is important to note a structural difference of the two variants of fusion. As explained before, the Euclidean distance variant enables the pair wise fusion of nodes of networks, so topology preservation is still valid in the fused network. This allows obtaining 2D maps such as the one showed in Fig. 4, which can be also easily obtained from single maps. This is impossible to do with the Voronoi similarity, as some neurons are not related to others and a position in the map in relation with the rest can not be determined

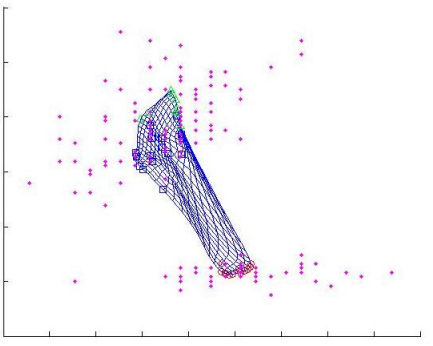


Fig. 1. A single ViSOM network represented over the iris dataset

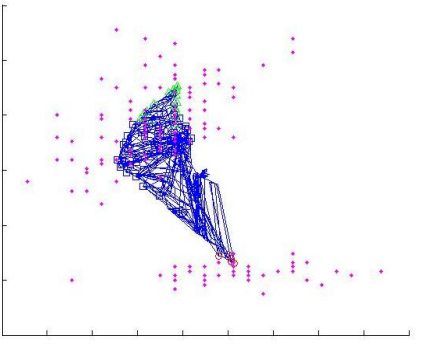


Fig. 2. The fusion of 5 ViSOM networks using the Euclidean Distance criterion

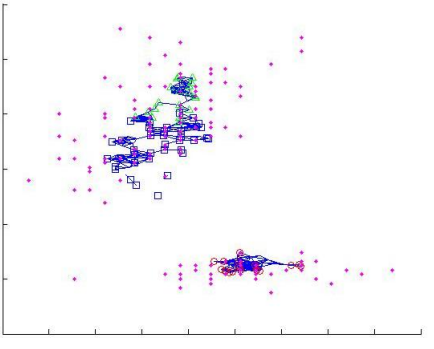


Fig. 3. The fusion of 5 ViSOM networks using the Voronoi polygon similarity criterion

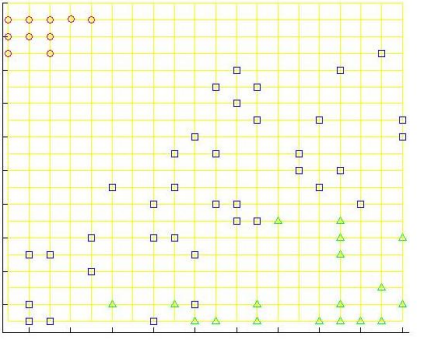


Fig. 4. The 2D map data representation of the fused network appearing on Fig. 2

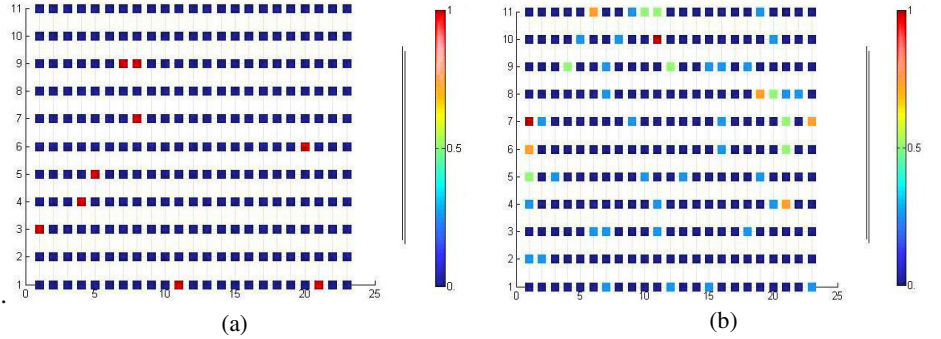


Fig. 5. Topographic error calculated over the single ViSOM of Fig 1 is shown in Fig 5(a) and over the Fusion by Euclidean distance of Fig 2 is shown in Fig 5 (b)

As reflected visually in Figures 1 and 2 the main problem of the Fusion by Euclidean distance is that it introduces distortions in the neighbourhood of the map. As can be seen in Figure 5a the single ViSOM includes few neurons with high topographic error that are located in a specific region of the map (coinciding with the gap between the linearly separable group and the other two), while in Fig 5b the number of neurons with a medium and high topographic are more numerous and more scattered along the whole map. This is a characteristic that should be corrected, maybe by a re-calculation of the neighbouring after the fusion.

Table 1. Comparison of the two topology preserving models using an ensemble of 10 maps to calculate the **MSQE** for: the average of all 10 maps, the fusion of the 10 maps using the distance criterion and the fusion of the maps using the Voronoi similarity criterion

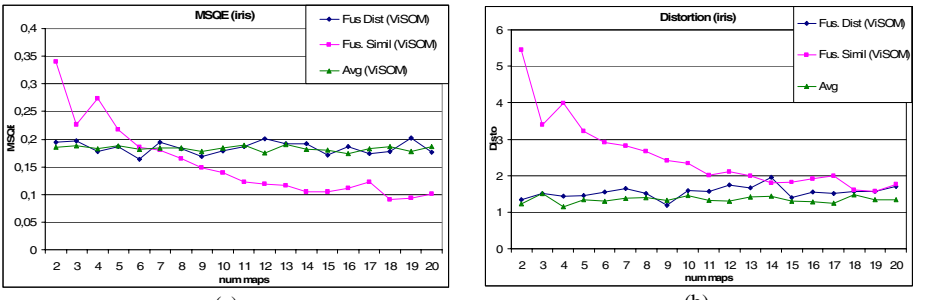
	SOM			***ViSOM***		
	Avg	*Fus. Dist*	*Fus. Simil*	*Avg*	*Fus. Dist*	*Fus. Simil*
Iris	0,196	0,200	0,142	0,183	0,179	0,139
Cancer	1.959	1.931	1.161	1,746	1,544	1,231
Wine	9,912	10,406	4,420	9,401	9,138	4,067

Table 2. Comparison of the two topology preserving models using an ensemble of 10 maps to calculate the **Distortion** for: the average of all 10 maps, the fusion of the 10 maps using the distance criterion and the fusion of the maps using the Voronoi similarity criterion

	SOM			***ViSOM***		
	Avg	*Fus. Dist*	*Fus. Simil*	*Avg*	*Fus. Dist*	*Fus. Simil*
Iris	1,354	1,500	2,127	1,451	1,593	2,336
Cancer	19,03	25,12	43,52	15,46	19,23	41,98
Wine	69,12	71,50	60,93	65,82	55,81	62,81

When comparing the two variants of the Topology preserving algorithms used, it can be inferred that the ViSOM obtains better results than the SOM, both for MSQE (Table 1) and Distortion (Table 2) measures, by a small margin, in the three datasets used. This is due to its updating of inter-neuron weights procedure, as it forces the map to adapt its inter-neuron distances to the inter-data distances of the input space; improving its adaptation to the dataset.

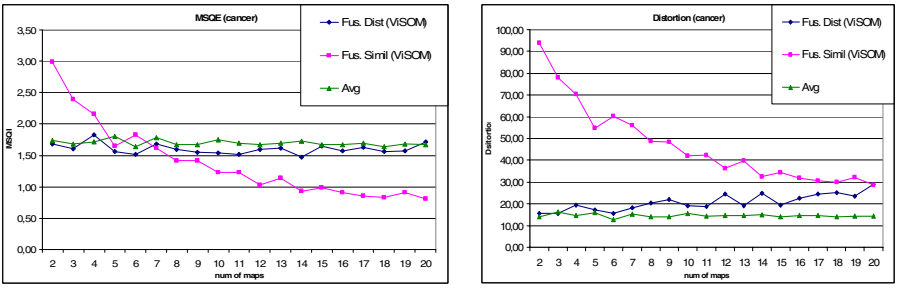
The comparative results obtained by the two fusion variants according to the number of networks trained in the ensemble, along with the average of the corresponding measures considering each network of the ensemble individually, are shown in Figures 6 and 7. In Fig 6 the results refer to the iris dataset, while Fig 7 represents the cancer dataset results. Regarding the MSQE measure (that is, how well the map units approximate the data entries in the dataset) it can be seen that the results for the fusion by distance are very similar to those of the average of the ensemble networks (even, in the cancer case, the error is consistently a bit lower). On the other hand, the fusion by similarity in Voronoi polygons obtains quite better results when



(a) (b)

Fig. 6. Results for the Iris Dataset using ViSOM maps. Fig 6(a) shows the MSQE of the different models (Fusion by Distance, Fusion by Voronoi Similarity and Average of Simple Model) according to the number of maps of the ensemble. Fig 6(b) shows the Distortion results for the same model. Both are calculated using the Iris dataset.

surpassing the number of 7 or 8 maps in the ensemble (in both iris and cancer datasets). This is due to the nature of the algorithm that enables the fused map to adapt better to the dataset by ignoring the neighbouring of the neurons that are not within the range of a certain threshold. So, while the final map does only keep the neighbouring between certain regions of the map (as it can be seen in Fig 3), this separated regions of the map can approximate better to data in the input space.



(a) (b)

Fig. 7. Results for the Cancer Dataset using ViSOM maps. Fig 7(a) shows the MSQE of the different models (Fusion by Distance, Fusion by Voronoi Similarity and Average of Simple Model) according to the number of maps of the ensemble. Fig 7(b) shows the Distortion results for the same model. Both are calculated using the Cancer dataset.

Regarding the distortion measure (that is taking into account the quality both of quantization and neighbourhood); the fusion by distance of neurons is quite similar to the average of the networks (although it is consistently a bit higher in both cases). Both the average and the fusion by distance have lower errors than the fusion by Voronoi similarity, due again to the way it preserves the neighbourhoods of units in the final map. A neuron is considered to be neighbour of all other neurons that are

within a certain distance. So while in a classic SOM with a square-shaped neighbouring a neuron can have between 4 and 2 neighbours (depending of its position in the map); in a fused SOM using the Voronoi polygons similarity a neuron does not have restrictions of the number of neighbours; altering the distortion of the final map. Despite this condition, when the number of neurons approximates to 20 the distortion becomes closer to that of the other two models. This could be due to the obvious increase in the number of neurons with the increase in the number of maps. Being more neurons to connect is more difficult to have isolated neurons or groups of neurons. In opinion of the authors of the paper, this phenomenon needs further study.

5 Conclusions and Future Work

The current work presents a study of a technique developed to obtain more reliable 2-dimensional maps representing multi-dimensional datasets. This is achieved by the use of the ensemble theory and a fusion process. Two different variants of the algorithm are considered and studied, by means of several widely used measures for quantifying the quality of this type of maps. It seems that the first presented option, the Euclidean distance criterion for fusion is the model that best preserves the main characteristics of the Self-Organizing Maps: both data quantization and topology preservation. Despite on this, the variant of the Voronoi polygon similarity has some interesting advantages that should be studied with the aim of being adopted to improve the final fused model.

Future work should include possible further study of the capabilities of each one of the variants, with the final objective of the proposition of a fusion algorithm developed to bring together the best characteristics of both variants. Also application and comparison of this new fusion algorithm with some other topology preserving ensemble models could be performed.

Acknowledgments

This research has been supported by the MCyT project TIN2004-07033.

References

1. Kohonen, T., Lehtio, P., Rovamo, J., Hyvarinen, J., Bry, K., Vainio, L.: A Principle of Neural Associative Memory. Neuroscience 2, 1065–1076 (1977)
2. Kohonen, T.: The Self-Organizing Map. Neurocomputing 21, 1–6 (1998)
3. Yin, H.: Visom - a Novel Method for Multivariate Data Projection and Structure Visualization. IEEE Transactions on Neural Networks 13, 237–243 (2002)
4. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms, p. 471210781 (2004)
5. Ron, M., Gunnar, R.: An Introduction to Boosting and Leveraging. In: Mendelson, S., Smola, A.J. (eds.) Advanced Lectures on Machine Learning. LNCS (LNAI), vol. 2600, pp. 118–183. Springer, Heidelberg (2003)

6. Yin, H.: Data Visualisation and Manifold Mapping Using the Visom. Neural Networks 15, 1005–1016 (2002)
7. Pölzlbauer, G.: Survey and Comparison of Quality Measures for Self-Organizing Maps. In: WDA 2004. Fifth Workshop on Data Analysis, pp. 67–82. Elfa Academic Press (2004)
8. Lampinen, J., Oja, E.: Clustering Properties of Hierarchical Self-Organizing Maps. Artificial Neural Networks 2, 1219–1222 (1992)
9. Ruta, D., Gabrys, B.: A Theoretical Analysis of the Limits of Majority Voting Errors for Multiple Classifier Systems. Pattern Analysis and Applications 5, 333–350 (2002)
10. Petrakieva, L., Fyfe, C.: Bagging and Bumping Self Organising Maps. Computing and Information Systems (2003)
11. Baruque, B., Corchado, E., Yin, H.: Visom Ensembles for Visualization and Classification. In: Sandoval, F., Gonzalez Prieto, A., Cabestany, J., Graña, M. (eds.) IWANN 2007. LNCS, vol. 4507, Springer, Heidelberg (2007)
12. Breiman, L.: Bagging Predictors. Machine Learning 24, 123–140 (1996)
13. Saavedra, C., Salas, R., Moreno, S., Allende, H.: Fusion of Self Organizing Maps. In: Sandoval, F., Gonzalez Prieto, A., Cabestany, J., Graña, M. (eds.) IWANN 2007. LNCS, vol. 4507, pp. 227–234. Springer, Heidelberg (2007)
14. Georgakis, A., Li, H., Gordan, M.: An Ensemble of Som Networks for Document Organization and Retrieval. In: AKRR 2005. Int. Conf. on Adaptive Knowledge Representation and Reasoning, p. 6 (2005)
15. Newman, D.J., Hettich, S., Blake, C.L. and Merz, C.J.: Uci Repository of Machine Learning Databases. University of California, Irvine, Dept. of Information and Computer Sciences (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html

Classification Based on the Trace of Variables over Time

Frank Höppner and Alexander Topp

University of Applied Sciences Braunschweig/Wolfenbüttel
Robert Koch Platz 10-14
D-38440 Wolfsburg, Germany

Abstract. To be successful with certain classification problems or knowledge discovery tasks it is not sufficient to look at the available variables at a single point in time, but their development has to be traced over a period of time. It is shown that patterns and sequences of labeled intervals represent a particularly well suited data format for this purpose. An extension of existing classifiers is proposed that enables them to handle this kind of sequential data. Compared to earlier approaches the expressiveness of the pattern language (using Allen et al.'s interval relationships) is increased, which allows the discovery of many temporal patterns common to real-world applications.

1 Introduction

In knowledge discovery, in particular in classification tasks, the values of some variables are used to predict the value of another variable, the so-called class variable or label. Classifiers are trained by presenting historical cases including the class value for training purposes. Each historical case usually represents a real world object at some specific point in time (usually the time of recording the case into the database). If the variables are volatile, however, then the class label may depend on the *history* or *development* of the variables rather than their value at the time of recording only. If measurements are numerical and taken periodically we obtain time series, which we may analyze directly or feed into a feature extraction process (such as the extraction of Fourier coefficients) and use these features in traditional classifiers. Problems arise, however, if the values are not measured at a constant rate, if there are gaps in the data or we have mixed data types (not only numerical). As an example from health care, we may measure a numerical ECG signal together with external influences such as medication.

Rather than using information from a single point in time we are interested in predicting the class by analyzing the complete history of the variables (such as a patient's file or log files for technical systems). The remainder of the paper is organized as follows: In the next section, we will motivate the kind of data representation that we are going to use, namely sequences of labeled intervals. In section 3 we give a brief overview of related work on the analysis of interval sequences in the literature and discusses drawbacks of the existing approaches.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 739–749, 2007.

Section 4 presents a new approach to this problem that greatly improves the expressiveness of patterns compared to earlier proposals. In section 5 we present some results from fault diagnosis and finish with the conclusions in section 6.

2 Representing Data Histories by Interval Sequences

In some cases, the state of an object in the real world changes monotonically. For instance, a customer purchases more and more products from a company, but usually does not send it back to the retailer at some later point in time. Thus, the set of purchased items grows but an item once bought is never removed from this set. In such a monotonic setting, it is sufficient to store when another product has been purchased. Time-stamped event sequences can be used to represent this kind of data.

If we consider objects of the real world in general, however, most of them do not disclose such a monotonic behavior. If we observe the operation of an electrical device, we observe changes of its internal state from 0 (off) to 1 (on) but also from 1 to 0. In a medical domain symptoms may occur and vanish again later on. The demand for a certain product may be overwhelming today but negligible some months later.

One could still use event sequences (one event indicating that the symptom occurs and another when it vanishes) to represent these non-monotonic changes of the world. Suppose we have three devices A, B and C and that we observe some malfunction but cannot tell exactly when and why the failure occurs. We have recorded several histories and want to induce knowledge about the circumstances of failure from this data. Let us assume that the problem arises if devices are turned on in the order A, B, C but turned off in the order A, C, B. The goal is to *discover* this pattern in our database of historical cases. When still using event sequences to represent this data, the target sequence is represented by the following sequence:

$$\mathbf{A}_{\text{on}}\ \mathbf{B}_{\text{on}}\ \mathbf{C}_{\text{on}}\ \mathbf{A}_{\text{off}}\ \mathbf{C}_{\text{off}}\ \mathbf{B}_{\text{off}}$$

Since the seminal work in [1, 2] many algorithms have been proposed for the analysis of event sequences. Our target sequence will be discovered by any of these approaches if it is found frequently in the histories and helps to discriminate the two classes (fault/no fault). In all approaches the sequence above is found being a subsequence of the following:

$$\mathbf{A}_{\text{on}}\ \mathbf{B}_{\text{on}}\ A_{\text{off}}\ \mathbf{C}_{\text{on}}\ A_{\text{on}}\ \mathbf{A}_{\text{off}}\ \mathbf{C}_{\text{off}}\ \mathbf{B}_{\text{off}}$$

However, this case does **not** correspond at all to our target pattern: It is not true that the devices are turned on in the order A, B, C, because A has already been turned off before C is turned on. Only the fact that A is reactivated later makes it possible to match the event sequence here. Thus, our quite simple target problem cannot be represented and matched correctly. The reason is that events in an event sequence are considered independent from each other, but in fact every 'power-on' event corresponds to the very next 'power-off' event referencing the same device. When expressing the temporal pattern as a sequence of events,

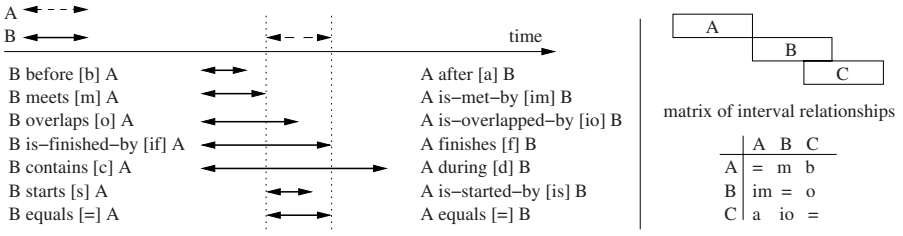
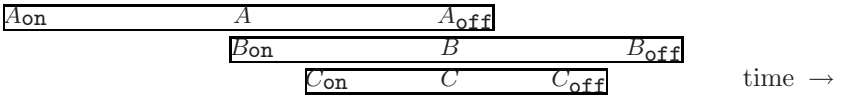


Fig. 1. There are 13 possible relationships between two intervals

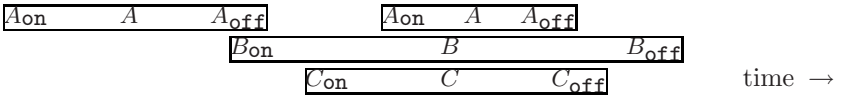
we would have to impose such constraints explicitly (but this is usually not done in the literature on event sequences).

The dependency between corresponding events is better prevailed if both events form a single data object rather than two independent ones: one interval for each up-time (and/or down-time) of the device. Now, using intervals the pattern looks like this:



Next, the notion of being a subsequence must be defined for interval sequences. Firstly, for P being a subpattern of sequence S all intervals in P must occur in S (just like with events). Secondly, the desired relationships between the intervals has to be formulated, just as the (partial) order in case of event sequences. Allen et al. [3] has introduced 13 interval relationships that are shown in Fig. 1. The relationship between intervals is more complex than that of events, but the concept remains the same: For any two intervals A and B the set of allowed relationships has to be specified. In our case, we could require the following pairwise relationships: A overlaps B and C, B contains C.

If we now revisit our second historical case, it is represented by the following interval sequence:



When trying to match the intervals in this sequence to our pattern we have two possibilities for A: the first A correctly overlaps B but does not overlap C and the second neither overlaps B nor C. So this notion of '*being a subsequence*' corresponds much better to our intention and does not mistake this sequence as an occurrence of target pattern.

This example consisted of dichotomous variables only, but variables at other scales can also be used (cf. Fig. 2). For instance, a new interval may be introduced whenever a categorical variable changes its value or a numerical variable changes its qualitative behavior (e.g. increasing/decreasing, low/medium/high, etc., see also [4]).

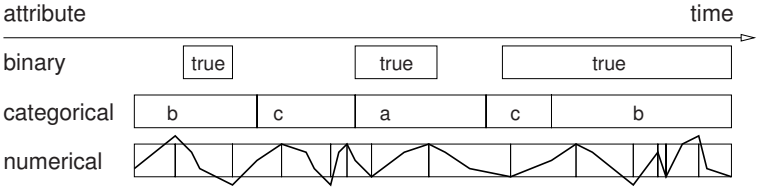


Fig. 2. Transformation of historical data into an interval sequence

3 Related Work

More formally, we define an interval sequence $s = (s_i, l_i, e_i)_{i=1..n}$ as a finite sequence of intervals $[s_i, e_i] \in \mathbb{R}^2$ with labels $l_i \in L$, where $i = 1..n$, $s_i < e_i$ and L is a finite set of labels. Our data set consists of several such sequences, each carrying an additional class label for classification. We denote the set of 13 interval relationships by $\mathcal{I} = \{b, a, m, im, o, io, if, f, c, d, s, is, eq\}$ (cf. Fig. 1).

In the literature there is comparatively little work on the analysis of interval sequences for the purpose of classification or knowledge discovery. Most of them have their own notion of a *pattern* (subpattern of an interval sequence) and how it is matched against another sequence. We can compare the different notions of a pattern using the following unifying view. A pattern P is captured by the following matrix, where the m rows and columns denote intervals (resp. their label) $k_j \in L$ and the matrix elements $R_{i,j} \subseteq \mathcal{I}$ the set of possible pairwise relationships:

	k_1	k_2	...	k_m
k_1	$\{eq\}$	$R_{1,2}$	...	$R_{1,k}$
k_2	$R_{2,1}$	$\{eq\}$	...	$R_{2,k}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
k_m	$R_{1,2}$	$R_{1,3}$	...	$\{eq\}$

On the diagonal, where we compare identical (sets of) intervals k_j, the relationship has to be *equals* ($R_{j,j} = \{eq\}$). A pattern P matches a sequence $S = (s_i, l_i, e_i)_{i=1..n}$ of length n, if we can find an injective mapping $\varphi : \{1, 2, .., m\} \to \{1, 2, .., n\}$ such that (a) label k_j in the pattern equals label $l_{\varphi(j)}$ in the sequence and (b) the interval relationship r between $[s_{\varphi(i)}, e_{\varphi(i)}]$ and $[s_{\varphi(j)}, e_{\varphi(j)}]$ is contained in $R_{i,j}$ for all $i, j \in \{1, 2, .., m\}$.

If $R_{i,j} = \mathcal{I}$ holds for all $i \neq j$, any interval relationship would suffice and matching of patterns reduces to an existence test for all interval labels (just as in association rule mining). If we restrict $R_{i,j}$ to a subset of $\{a, b, eq\}$ we can simulate event sequences where two events either occur in parallel or in succession. Let us now consider how some approaches to the analysis of interval sequences fit in this view:

– In [5] the relation between k_1 and k_2 is always unique ($|R_{1,2}| = 1$). For any further interval k_j the relationship is again unique with respect to the

smallest interval that encloses all preceding intervals $k_1 ... k_{j-1}$, but this leaves several possibilities open for the individual interval comparison $R_{i,j}$. As an example, if $R_{1,2} = \{o\}$ and the union of k_1 and k_2 again *overlaps* k_3 we have $R_{1,3} = \{o, m, b\}$ and $R_{2,3} = \{o, s, d\}$.
- In [6] the relation $R_{i,j}$ is either *contains*/*during* or remains unspecified.
- In [4] all possible relationships $R_{i,j}$ are exploited, given that $|R_{i,j}| = 1$.
- Other authors prefer other relationships over Allen et al.'s, such as *is-older-than* or *survives* [7]. However, since the 13 relationships by Allen et al. are complete, those relationships can always be represented as a disjunctive combination of relationships in $\mathcal{I}$.

The approach in [4] is quite general in the sense that *all* $R_{i,j}$ are configurable whereas the other approaches focus either on a few $R_{i,j}$ or on a specific subset of $\mathcal{I}$ only. On the other hand, *all* pairwise relationships are strictly fixed in this approach, but many relationships that occur frequently in practice cannot be expressed by a single relationship $r \in \mathcal{I}$. For instance, '*A starts within B*' clarifies the location of the start time A_{on} of A with respect to B ($A_{\text{on}} \in]B_{\text{on}}, B_{\text{off}}[$), but the position of A_{off} is left open. Therefore, a number of different interval relationships between A and B are possible: *A is-overlapped-by B*, *A during B*, *A finishes B*. To reflect the true relationship correctly, $R_{i,j}$ should be chosen as $\{io, d, f\}$. Another example is '*A and B are disjoint*' that can be resembled by *after* and *before* relationships ($R_{i,j} = \{a, b\}$).

4 Increasing the Expressiveness

The goal of this work is to relax the conditions on the sets of interval relationships $R_{i,j}$ to increase the expressiveness of the pattern language and thus to discover more realistic dependencies. We want to achieve this in two steps:

Firstly, rather than specifying *all* relationships between any two intervals, we want to restrict ourselves to those relationships that are really helpful for the classification. By reducing the number of constraints in this way, the patterns become more robust against effects of dilation and translation and an undesirable rule fragmentation is avoided. As an example, suppose the true relationship is '*A and B occur before C*', which is quite a simple pattern. It does not contain any information about the relationship between A and B, so all 13 relationships are possible. The approach of [4] is capable of learning the relationship '*A before C*' and '*B before C*' but also requires a single relationship between A and B. As we have seen, there is no such single relationship, so if we are forced to fix this relationship artificially, the support of this pattern is fragmented into 13 rules. A rule evaluation measure would rank each of these 13 rules much lower than a rule that corresponds to the true pattern.

Secondly, we want to relax the requirement $|R_{i,j}| = 1$ such that relationships beyond the basic ones can be discovered (e.g., *does not intersect*: $R_{i,j} = \{$*after*, *before*$\}$).

4.1 Choice of the Classifier

We use a standard rule (or tree) learner that develops its model incrementally, such as ITRule [8] or C4.5 [9]. Such algorithms have already been used for temporal data in the literature, but there a variable is often just replicated several times to represent its value at different points in time. Since these points in time have to be fixed before the model is learned and deployed, a dilation or translation of events may easily misguide such a classifier.

The set of available variables for rule development is therefore altered in our approach and additionally depends on how far a rule has already been developed. At the beginning, one dichotomous variable per interval label is introduced. At this stage, the rule learner derives rules that consist of conditions on the existence or absence of certain intervals (labels) in the complete history, such as:

$$\textbf{if } \neg D \wedge A \wedge B \textbf{ then } \text{fault} \tag{1}$$

(reading 'if there is no interval labeled D but two intervals labeled A and B in the history, then we predict fault'). As soon as a rule requires the existence of at least two intervals in the history (here: A and B), additional variables are provided for further rule refinement. For each pair of such intervals a nominal attribute is introduced over the domain $2^{\mathcal{I}}$, indicating all relationships between A and B found in the history. In the next step the rule may now be specialized by adding yet another clause on the existence or absence of an interval but also by adding a condition on the relationship between two intervals, such as:

$$\textbf{if } \neg D \wedge A \wedge B \wedge A\langle R\rangle B \textbf{ then } \text{fault} \tag{2}$$

How to find this set R will be discussed in the next section. For instance, if $R = \{m, o\}$ the rule holds only for those cases where A *meets* or *overlaps* B. If a rule does not contain any condition on the relationship between two intervals k_i and k_j, this situation of complete ignorance corresponds to $R_{i,j} = \mathcal{I}$.

4.2 Sets of Relations

The newly introduced variables over the domain $2^{\mathcal{I}}$ have 2^{13} different values. For obvious reasons we do not want to check each of the 8192 possibilities to find the best rule specialization. To reduce this number we form groups of interval relationships and treat the relationships in a group collectively. If the intervals are derived in a preprocessing step from some other data, the exact location of the interval bounds is quite often questionable due to the several (heuristic) steps applied during preprocessing. In such a case it does not make much sense to consider relationships that require an exact match of interval bounds in full detail, because these matches are rather incidental. Here we are only interested in the relationships b, a, o, io, d, c but for reasons of completeness we form groups with the remaining relationships such that we obtain a partition such as:

$$\mathcal{G} = \{\{b, m\}, \{a, im\}, \{o\}, \{io\}, \{d\}, \{c\}, \{eq, s, is, f, if\}\} \tag{3}$$

While it is quite canonical to group m and b, it is less obvious what to do with *is* or *if*, for instance. Grouping $\{o, if\}$ and $\{io, f\}$ also appears to be a reasonable choice. For the remainder of the paper the actual partition is not important, it may be chosen differently from application to application.

Rather than considering all 2^{13} subsets of $\mathcal{I}$ we restrict ourselves to subsets of $\mathcal{G}$. More precisely, for $R_{i,j}$ we consider only subsets of $\mathcal{I}$ that can be represented as the union of elements of $\mathcal{G}$. For the remainder of this section by 'relationship r' we refer to *a group of relationships* $r \in \mathcal{G}$ rather than single element of $r \in \mathcal{I}$. Suppose that we have found a rule (1) requiring the existence of two intervals, say A and B. We want to extend the rule in the fashion of example (2), so our task is to find the set of relations R leading to the best refined rule according to some rule evaluation measure (e.g. the J-measure used in [8]). All necessary information for the rule evaluation measure can be found in the following contingency table, where the left/right part corresponds to the rule before/after the refinement on the interval relationship.

	base rule (1)		refinement (2)	
number of cases	class		class	
	positive	negative	positive	negative
rule antecedent holds	a	b	p_R	n_R
rule antecedent does not hold	c	d	$P - p_R$	$N - n_R$
total	P	N	P	N

Since the *base rule* is already given, the left half of the table is already known. The total number of positive and negative cases (P and N, resp.) remains identical for the base rule and the refinement, therefore it is sufficient to determine p_R and n_R for each possible refinement. To decide which subset R improves the given rule best, we need to construct the right contingency table for each possible choice of R. Thanks to the introduction of groups we have $2^7 = 128$ possibilities for R. This still requires some effort that we want to reduce further.

We restrict our discussion to the determination of n_R, but the same arguments hold analogously for p_R. While n_R denotes the number of sequences in which A and B can be observed in (at least) **one** of the relationships in R, we use m_R to denote the number of sequences in which A and B can be observed in **all** of the relationships in R simultaneously. Let us now examine how to determine n_R for the various choices of R:

- $|R| = 1$: We determine $n_{\{r\}}$ directly, that is, among the sequences that satisfy the antecedent of the base rule, we count in how many cases A and B satisfy relationship r.
- $|R| = 2$: Due to $|A \cup B| = |A| + |B| - |A \cap B|$ we have $n_{\{r,s\}} = n_{\{r\}} + n_{\{s\}} - m_{\{r,s\}}$. Since we know already the frequencies $n_{\{r\}}$, we additionally determine $m_{\{r,s\}}$ for all pairs of relationships r, s.
- $|R| = 3$: Repeating the same argument of the previous case, we have

$$n_{\{r,s,t\}} = n_{\{r\}} + n_{\{s\}} + n_{\{t\}} - m_{\{r,s\}} - m_{\{r,t\}} - m_{\{s,t\}} + m_{\{r,s,t\}}$$

We only *approximate* n_R for this case, by neglecting the term $m_{\{r,s,t\}}$. We consider the error we introduce by this estimation to be rather small, because $m_{\{r,s,t\}}$ counts how often intervals A and B have been observed in *all three relationships r, s and t in the same sequence*.[1]

– $|R| > 3$: Instead of considering sets R with $|R| > 3$ we take $S = \mathcal{G} \backslash R$ into account, leading to $|S| \leq 3$. For two given intervals A and B, the condition $A\langle R\rangle B$ is equivalent to $\neg(A\langle S\rangle B)$. Therefore using the negation of a 'small' set S (of size ≤ 3) can be a substitute for using a 'large' set R (of size ≥ 4).[2] The frequency of $\neg(A\langle S\rangle B)$ can be readily obtained from $b - n_S$ (cf. contingency table), because b denotes the number of histories before rule refinement and n_S is the number of cases for which one relation $r \in S$ has been observed – consequently in $b - n_S$ cases no relationship $r \in S$ has been found in the sequences (but a relation $r \in R$).

By counting the frequencies n_R for $|R| = 1$ (7 counters when using (3)) and m_R for $|R| = 2$ (21 counters) we have reduced the number of frequency counts from 128 to 28. Only for sets R with cardinality 3 and 4 we thereby introduce a small error, in all other cases we obtain exact results.

Being able to derive the contingency table from these frequencies, we could in principle check for each possible value of R how the rule measure of the respective algorithm evaluates the refinement. Rather than trying all possibilities we use the following heuristic: The original rule is evaluated by some rule evaluation measure and yields a initial value of J_0 that any refinement must beat. We start from an empty set $R_0 = \emptyset$. For a given set R_i we choose the most promising relationship r such that the rule measure is optimized by $R_{i+1} = R_i \cup \{r\}$. This *optimization* is achieved for most rule measures by maximizing the difference $p_R - n_R$ from the contingency table. Among the sets of relationships $R_1 \subseteq R_2 \subseteq R_3$ (and $S_1 \subseteq S_2 \subseteq S_3$ for the negations) we finally choose the one that is evaluated best by the rule measure. In case this is J_0, no rule refinement is made. This represents, of course, a greedy optimization that does not guarantee global optimality.

5 Evaluation

Future work will include the application of this algorithm to medical data from health care. In this section, we apply our technique to a simple case in fault diagnosis. Figure 3 shows a small Java program using two parallel threads accessing a shared variable. The access to this variable is not synchronized, which is why some undesired effects occur. The first dummy thread calls methods f, g and h, one after the other. f stores a first value in the shared variable, g a second one,

[1] If, however, this assumption is violated, we also determine $m_{r,s,t}$.

[2] While this is perfectly true for two given intervals, when arguing about the relationship of A and B in a set of sequences there are little differences in the semantics, because multiple instances of A and B may occur.

```
class A implements Runnable {
  Random random = new Random();
  private int[] shared;
  boolean ok;
  A(int[] data) { shared=data; }
  void f(int i) {
    Thread.sleep(random.nextInt(40));
    shared[0]=i;
  }
  void g(int i) {
    Thread.sleep(random.nextInt(40));
    shared[1]=i;
  }
  int h() {
    Thread.sleep(random.nextInt(40));
    int r = shared[0]+shared[1];
    return r;
  }
  public void run() {
    Thread.sleep(random.nextInt(40));
    int i,j;
    f(i=random.nextInt(10));
    g(j=random.nextInt(10));
    ok = (i+j==h()); // class label
  }
}
```

```
class B implements Runnable {
  Random random = new Random();
  private int[] shared;
  B(int[] data) { shared=data; }
  void k() {
    int i = shared[1];
    shared[1] = random.nextInt(1000);
    Thread.sleep(random.nextInt(40));
    shared[1] = i;
    return i;
  }
  public void run() {
    Thread.sleep(random.nextInt(100));
    k();
//  Thread.sleep(random.nextInt(40)); k();
  }
}
public class Demo {
  static int[] data = new int[2];
  public static void main(String[] args){
    ExecutorService app =
      Executors.newFixedThreadPool(2);
    app.execute(new A(data));
    app.execute(new B(data));
    app.shutdown();
  }
}
```

Fig. 3. Example Source Code from which interval sequences were generated: Calls to methods f, g, h or k are recorded, the history of a complete run is labeled with the value of the boolean variable 'ok' in class A

and h reads both values and returns their sum. We compare if the returned sum is correct (which gives us the classification label for a test run of the program). A second thread consists of a single call of k that also uses the shared variable temporarily, but at the end it restores the value from the beginning. A faulty result occurs, for instance, if the call of g overlaps the call of k. This program has been executed several times: each function call gives us an interval in our interval sequence (the label is the name of the method, the execution time gives the temporal interval).

To demonstrate the advantage of the proposed approach, the following table shows the *best* rule obtained when only single relationships ($|R| = 1$) are allowed (J-value of 0.1). A critical situation is given, if k covers the endpoint of either g or h. While this cannot be expressed exactly using a single relationship only, it is very well recovered from the top two rules delivered by the proposed algorithm – a great improvement in terms of the J-value is achieved. The third rule illustrates

the usefulness of relationship negation: A fault will **not** occur if f occurs *before* k, which is expressed by the negative condition in the third rule.

single relation only	**if** $k \wedge g \wedge g\langle\{o\}\rangle k$ **then** fault	J=0.100
any number of relations	**if** $k \wedge g \wedge g\langle\{o\},\{d\}\rangle k$ **then** fault	J=0.205
	if $k \wedge h \wedge h\langle\{o\},\{d\}\rangle k$ **then** fault	J=0.139
	if $k \wedge f \wedge \neg(f\langle\{b,m\}\rangle k)$ **then** fault	J=0.078

A second dataset includes the interval k twice (uncomment last line of B.run()) and we generate rules that predict *no-fault* rather than *fault*. The best rule of our approach has a J-values more than twice as high as the best rule obtained when single relationships are allowed only. The first rule corresponds quite well to a negation of the top rule in the previous table. The second rule is an example for a rule that refines multiple relationships.

single relation	**if** $k \wedge g \wedge g\langle\{o\}\rangle k$ **then** fault	J=0.043
multiple relationships	**if** $g \wedge k \wedge \neg(k\langle\{io\},\{c\},\{eq,s,is,f,if\}\rangle g)$ **then** no-fault	J=0.091
	if $h \wedge f \wedge k\langle\{b,m\},\{o\}\rangle f \wedge k\langle\{b,m\}\rangle h$ **then** no-fault	J=0.074

6 Conclusion

Most of the work on mining sequential data deals with event sequences. We have argued in this paper, why interval sequences are better suited to capture the trace of variables over time. The few papers on mining interval sequences in the literature make use of Apriori-like algorithms. In this paper we have proposed a way to extend more traditional classifiers (such as rule or tree learners) to handle interval sequences. The resulting models are well suited to express temporal dependencies that are common in realistic applications, the expressiveness is greatly improved over earlier approaches.

References

[1] Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proc. of 11th Int. Conf. on Data Engineering, Taipei, Taiwan, pp. 3–14 (March 1995)

[2] Mannila, H., Toivonen, H., Verkamo, A.I.: Discovering frequent episodes in sequences. In: Proc. of the 1st ACM SIGKDD Int. Conf. on Knowl. Discovery and Data Mining, Menlo Park, Calif., pp. 210–215 (1995)

[3] Allen, J.F., Kautz, H.A., Pelavin, R.N., Tenenberg, J.D.: Reasoning about Plans. Morgan Kaufmann Publishers, San Francisco (1991)

[4] Höppner, F.: Discovery of temporal patterns – learning rules about the qualitative behaviour of time series. In: Siebes, A., De Raedt, L. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 192–203. Springer, Heidelberg (2001)

[5] Kam, P.S., Fu, A.W.C.: Discovering temporal patterns for interval-based events. In: Kambayashi, Y., Mohania, M.K., Tjoa, A.M. (eds.) DaWaK 2000. LNCS, vol. 1874, pp. 317–326. Springer, Heidelberg (2000)

[6] Villafane, R., Hua, K.A., Tran, D., Maulik, B.: Knowledge discovery from series of interval events. Journal of Intelligent Information Systems 15(1), 71–89 (2000)
[7] Freksa, C.: Temporal reasoning based on semi-intervals. Artificial Intelligence 54(1), 199–227 (1992)
[8] Smyth, P., Goodman, R.M.: An information theoretic approach to rule induction from databases. IEEE Trans. on Knowledge and Data Engineering 4(4), 301–316 (1992)
[9] Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco (1993)

Extracting Meaningful Contexts from Mobile Life Log

Youngseol Lee and Sung-Bae Cho

Dept. of Computer Science, Yonsei University
Shinchon-dong, Seodaemun-ku, Seoul 120-749, Korea
tiras@sclab.yonsei.ac.kr,
sbcho@cs.yonsei.ac.kr

Abstract. Life logs include people's experiences collected from various sources. It is used to support user's memory. There are many studies that collect and store life log for personal memory. In this paper, we collect log data from smart phone, derive contexts from the log, and then identify which is meaningful context by using a method based on KeyGraph. To evaluate the proposed method, we show an example of the meaningful places by using contexts and GPS logs collected from two users.

1 Introduction

Several researchers have recently developed life logging system. Life log includes people's experiences which are collected from various sensors and stored in mass storage device. It is used to support user's memory and satisfy user's needs for personal information. If he wants to inform other people of his experience, he can easily share his experience with them by means of providing his life log.

Here, we use smart phone for collecting life log (GPS, SMS, call, charging, MP3, photos taken, images viewed, and weather information). Smart phone is a mobile device that includes color LCD screen, mass storage, large memory, and communicative function by using Wi-Fi, Bluetooth, and infrared. It also has a variety of software such as scheduler, address book, media player, and e-book. Mika Raento developed a framework for collecting contexts from smart phone [1], which collects GSM Cell ID, Bluetooth, GPS data, phone data, SMS data, and media information that are transmitted to the server. The contexts could be provided for other contents as additional information. Panu collects log data from mobile devices, and extracts features by pre-processing the log data [2]. The mobile device uses GPS log, microphone, temperature, moisture, and light sensor.

In life logging system, data mining is one of important issues, which identifies more important information than others. Life log system collects all information related to user's experience, which includes more data as time passes. Therefore, it is practically difficult to explore and search large amount of life log. It would take enormous time and be boring to explore all life log data.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 750–759, 2007.

In order to prevent from wasting time, it should be possible to check the priority of data and explore more important data. If enormous amount of data are collected from various sensors, it would be effective to explore only meaningful data.

In this paper, we collect log data from smart phone, derive contexts from the raw log, and then identify which is more meaningful context. To evaluate the proposed method, we show an example of the meaningful places by using contexts and GPS log data.

2 Related Works

Context-based video retrieval system recorded life-log video with contexts, and used contexts to search parts of video [3]. User could query related to contexts, and then the system response the candidates of video data. However, if video data are accumulated for a long time and there are too many video data, user would have difficulty to search necessary part of video. It would be more helpful to extract the candidates of meaningful contexts.

MyLifeBits Project is one of the implementations of personal record database system [4]. Personal information is collected by PC, SenseCam and so on, and stored in MS SQL server with relationships among personal information. However, user would have difficulties to explore and search contents because of large amount of personal data.

KeyGraph-based mobile contents management system was suggested to manage user's information in mobile device, which extracted important information using KeyGraph algorithm and provided searching or exploring contents [5]. The problem of the system is using only log data. If analysis and inference of the data was added to the system, it would give better performance.

3 Proposed Method

The whole system consists of four parts: collecting logs, preprocessing logs, inferring contexts using Bayesian networks, and data mining, as shown in Figure 1.

3.1 Log Collection

In this paper, logs are collected from smart phone. The definition of logs is user's all information collected from smart phone. The kinds of logs are GPS, SMS, call, MP3, photo, viewer, charge information from smart phone, and weather information from Internet. Table 1 summarizes the information that could be collected from smart phone. Places visited by user are extracted from GPS log. Records for telecommunication are extracted from call and SMS log. Weather information from Internet can affect user's emotion and status. MP3 music affects user's emotion. Memorable event is derived from photo record. Record of image viewer provides information about user's favorite photo or picture.

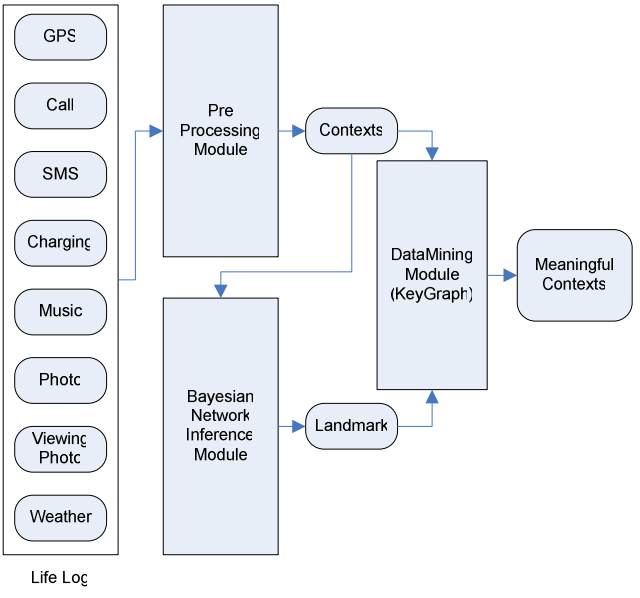


Fig. 1. System overview

Table 1. Logs collected from smart phone

Log Type	Information from Log
GPS	Longitude, Latitude, Speed over ground, Course over ground, Date, Time
Call	Phone number, StartTime, EndTime, Transmission/Receipt/Absence
SMS	Phone number, Send/Receive, Time
Picture Viewer	FileName, StartTime, EndTime
Photo	FileName, GenerationTime
Weather	Weather, Distance (km), Cloud (%), Temperature (°C), discomfort index (%), sensory temperature (°C), Rainfall (mm), Snowfall (cm), Humidity (%), the direction of the wind, wind speed (m/s), sea-level pressure (hPa)
MP3	Title, StartTime, EndTime, Position
Charging	Percentage of charging, Charging status, Time

Table 2 shows the period of collecting logs. Period of collection varies according to log type. Viewing picture and listening MP3 music are collected at execution of the application. Log data of SMS, call, photo are collected once for a day. GPS location data and information of charging are collected per thirty seconds.

Table 2. Collection of log data and period of collection

Log Type	Collection of Log	Period of Collection
GPS	Collection from GPS module	per second
Call	Collection from stored records for telecommunication	per day
SMS	Collection from stored records for SMS messages	per day
Picture Viewer	Collection from records for using image viewer	per execution of image viewer
Photo	Collection from records for generation of photo file	per day
Weather	Collection from Internet	per day
MP3	Collection from MP3 player	per execution of MP3 player
Charging	Collection from background program	per 30 seconds

3.2 Log Preprocessing and Context Generation

Some information must be transformed to semantic information. For example, GPS coordinate is a real value composed of longitude and latitude. Its value is not intuitive to the human, and so it is transformed to nominal value, such as school, hospital, etc. Similarly, other value is transformed from real value to nominal value. Listening one song more than 5 times can be transformed to 'listening frequently one song'. This transformation is required to input the context to Bayesian network inference module for landmark.

In low level data, frequent or rare data could be meaningful to user. Therefore, statistical analysis of the following items is executed.

- Daily frequency of events
- Sum of daily events
- Elapsed time from the last events occurred.

Table 3. Increase and Decrease of Density

Log Type	Increase of Density	Decrease of Density
GPS	If (GPS event) density++	If (one hour after the event) density--
Call	If (Call event) density++	If (one hour after the event) density--
SMS	If (SMS event) density++	If (20 minutes after the event) density--
Viewing Photo	If (Seeing photo event) density++	If (5 minutes after the event) density--
Photo	If (Photo event) density++	If (30 minutes after the event) density--
Weather	No density	No density
MP3	If (Listening music event) density++	If (30 minutes after the event) density--
Charging	No density	No density

Density of the events is also meaningful to user. Table 3 shows the criteria of increase or decrease of density of the events. If events occurred continuously, the value of density gets increased, leading to the moment of high density of events. The additional information of logs, frequency, density, etc is context. These contexts could be used for user modeling or providing service.

3.3 Landmarks Derived from Contexts

Contexts can be useful basis of inference of landmarks that are semantic information or events to remain user's memories. In order to infer appropriate landmarks, Bayesian network module is used. In our previous work, we attempted to extract landmarks through Bayesian network inference module [6], which made an inference of user's emotions and activities according to various life log captured from mobile devices. In this paper, they are used to make context-clusters because landmark represents the major change of contexts.

3.4 Data Mining by KeyGraph

In this paper, KeyGraph is selected to extract the meaningful contexts and the relations among contexts from the context-groups. Probability model or ontology and other rule-based methods can be used to distinguish meaningful contexts from whole contexts. These methods are using frequency or probability of context based on heuristics in order to estimate context. They shall miss infrequent and unknown but personally meaningful contexts. KeyGraph is a chance discovery algorithm that can estimate the importance of context by considering relationships of contexts. Each segmented context-group contains contexts that share user's common status, and clusters are formed from the contexts that frequently share user's common status. Contexts in a cluster have common characteristics. If a context links to more than two clusters, it is regarded as a turning point in user's contexts. The context is meaningful in user's life.

Frequent contexts are selected, and then the relationship among contexts is calculated by summing up the frequencies of contexts in a context-group. $|X|_S$ is a frequency of a context X in a context group S, and $relationship(w_i, w_j)$ is the strength of relationship between contexts w_i and w_j.

$$relationship(w_i, w_j) = \sum_{S \in D} \min(|w_i|_S, |w_j|_S) \tag{1}$$

Links that connect between the contexts are selected in strong relationship ($relationship(w_i, w_j)$) order. If all frequent contexts cannot be connected, the graph structure is divided into more than two clusters. If a context links to more than two clusters, it is regarded as a meaningful context.

clusterOne in equation (2) is the sum of the strength of relationships between a context w and a cluster g.

$$clusterOne(w, g) = \sum_{S \in D} |w|_S |g - w|_S \tag{2}$$

clusterAll in equation (3) is the sum of the strength of relationships between a context *w* and all clusters in all segmented place-sequences.

$$clusterAll(g) = \sum_{S \in D} \sum_{w \in S} |w|_S |g - w|_S \tag{3}$$

$$|g - w|_S = \begin{cases} |g|_S - |w|_S (w \in g) \\ |g|_S (w \notin g) \end{cases} \tag{4}$$

clusterOne and *clusterAll* are needed for *clusterValue* in equation (5), which is the probability of the strength of relationship between a context and a cluster.

$$clustervalue(w) = 1 - \prod_{g \subset G} (1 - \frac{clusterOne(w, g)}{clusterAll(g)}) \tag{5}$$

The Higher *clusterValue* of the place is, the higher the importance of the context is in all place-sequences. Finally, the contexts that have high *significance* value in equation (6) are selected as meaningful contexts.

$$significance(w_i, w_j) = \min(|w_i|_S, |w_j|_S) \tag{6}$$

where w_i is a place of high frequency value, and w_j is a place of high cluster value.

4 Experimental Results

In order to evaluate the usefulness of the proposed method, we show the extraction of meaningful places from visited places by using GPS logs, contexts and landmarks that two undergraduate students collected for two weeks. Location is one of important contexts for many systems. For example, life-log video system records video data by using a small wearable camera [7], which stores location as an index to retrieve necessary video data. MyLifeBits collects quite a lot of information about a person and constructs personal database [4], which also collects location as one of useful information. Therefore, we extract meaningful places from mobile life log as an example of extracting meaningful contexts.

4.1 Log Collection

We used logs that two undergraduate students had collected for two weeks to evaluate the proposed method. Figure 2 depicts the number of records in the log data.

We also requested that users report their life. It is required that users record their visiting places, visiting frequencies, and their activities. The user does not usually memorize all places for daily life. If user records one place in his report, the place is the memorable and meaningful place for him. Therefore, the places extracted from the proposed method should be compared with reports. Other contexts except place need the same criteria.

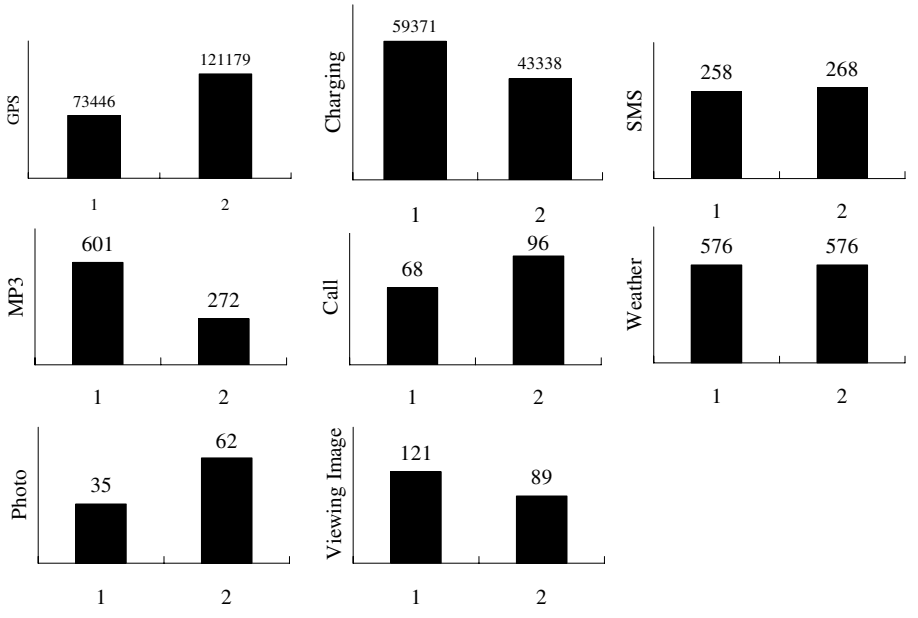


Fig. 2. Logs from two undergraduate students

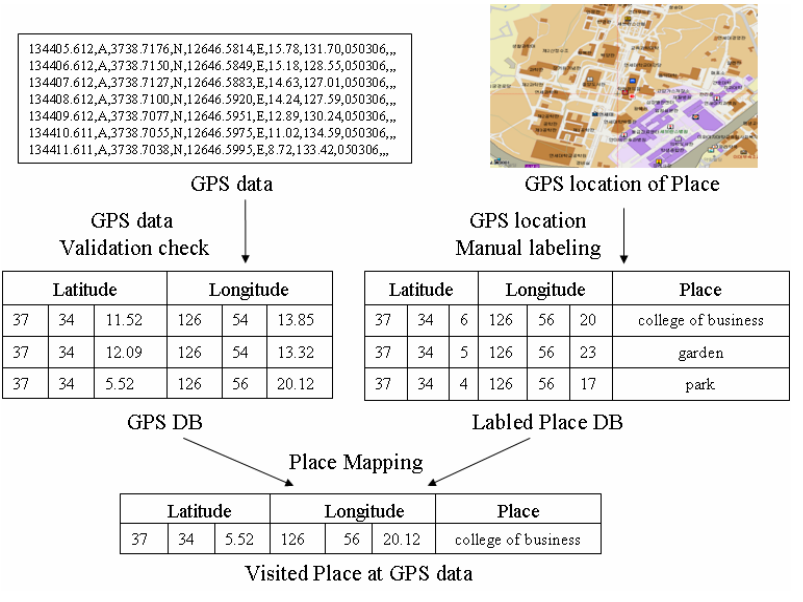


Fig. 3. Labeling GPS data and converting to place name

4.2 Log Preprocessing and Context Generation

Our system uses log data collected from smart phones. GPS data is numerical that contains longitude and latitude. Getting visited places needs the process that converts GPS data to place information. Figure 3 shows the process. In addition, all logs are required to analyze frequency and density, and so on.

4.3 Inference from Contexts

Landmarks are extracted from 39 Bayesian networks. 19 of them are designed for place/activity inference, 13 are for user's emotion/condition inference, 5 are for circumstance situation inference, and 2 are for event inference [6]. The total number of landmarks extracted from 2 undergraduate students is 1522. But not a few of them coincided with each other, so that the number of sentences is quite fewer than that of landmarks.

4.4 Data Mining

In order to define the problem clearly, only contexts of place and landmarks are used to extract meaningful places from all contexts and landmarks. Polaris [8] is used to visualize the results. The result of running KeyGraph of student 1 is like Figure 4. {university map, Yonsei engineering research center, Industry-University research center, soccer stadium, movie theater, physical education center, Yonsei eagle statue, amphitheater} are relatively frequent places. {department store, bridge, college of liberal art, college of theology, laboratory} are selected as rare and meaningful places for student 1.

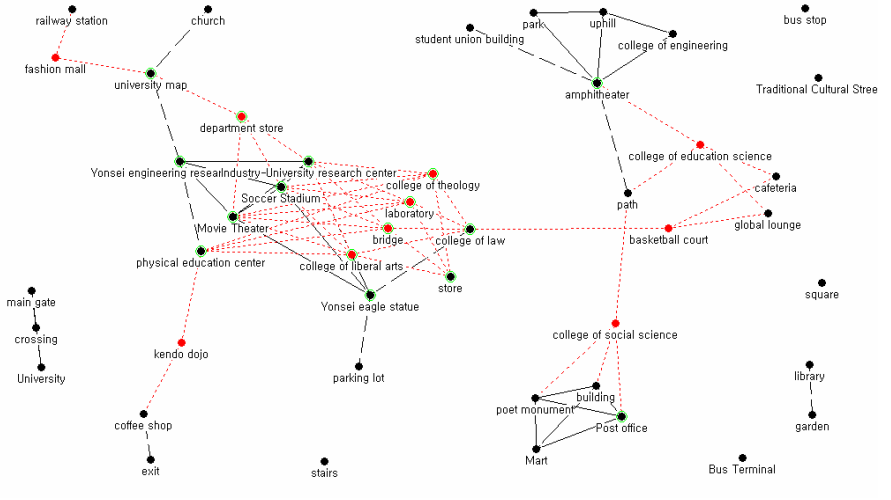


Fig. 4. Graph generated with places (student 1)

'department store' and 'laboratory' are represented only two times in data samples, but visiting department store is meaningful event in shopping, and laboratory is an important place for our experiment in analyzing her log data. We can find rare and meaningful places.

Student 2 is similar to student 1. Student 2 provides the most GPS data set. Figure 5 shows the result. {parking lot, Yonsei engineering research center, college of engineering, cinema, coffee shop, Chinese restaurant, square, crossing, global lounge, flower garden, college of music, main auditorium, amphitheater} are relatively high frequency places, and {department store, electronic market} are key places. Especially, two places (department store and electronic market) are selected as key places. These places are related to shopping. Department store is related to usual shopping, and electronic market is related to buying a cell-phone with her boyfriend. They are meaningful places visited in two weeks.

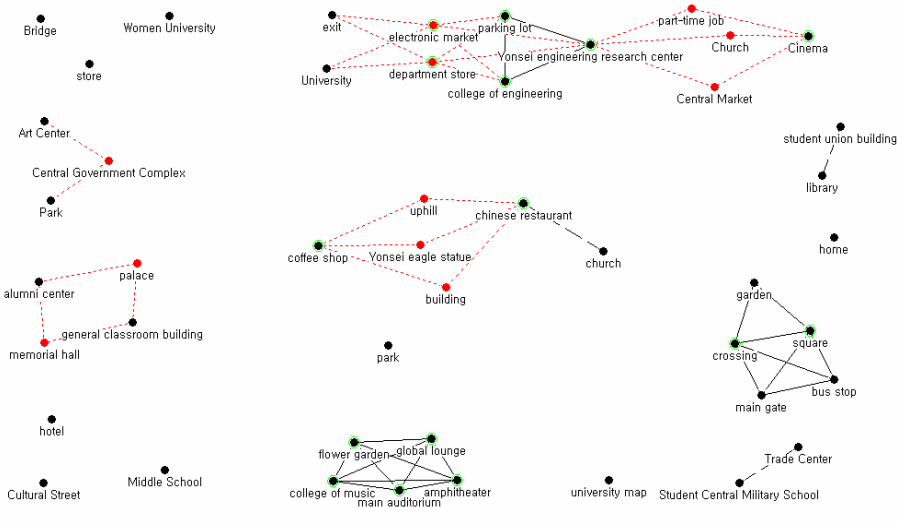


Fig. 5. Graph generated with places (student 2)

5 Summary and Future Work

We have proposed a method to extract meaningful contexts from life logs using KeyGraph with landmarks. Here, the landmarks are extracted through Bayesian network inference module and represents user's action or emotion in real life. In order to evaluate the proposed method, it is applied to a sequence of places visited by two undergraduate students for two weeks. As a result, it can extract meaningful places for users. What remains to be studied are collecting more data from more students and extracting more meaningful one from other contexts, such as SMS texts, MP3 use, Call log, photos taken, and so on.

Acknowledgments. This work was supported in part by LG Electronics and MIC (Korea) under ITRC.

References

1. Raento, M., Oulasvirta, A., Petit, R., Toivonen, H.: ContextPhone: A Prototyping Platform for Context-aware Mobile Applications. IEEE Pervasive Computing 4(2), 51–59 (2005)
2. Panu, K., Jani, M., Juha, K., Heikki, K., Esko-Juhani, M.: ContextPhone: Managing Context Information in Mobile Devices. IEEE Pervasive Computing 2(3), 42–51 (2003)
3. Aizawa, K., Hori, T.: Context-based Video Retrieval System for the Life-log Applications. In: Proc. of MIR 2003, pp. 31–38. ACM, New York (2003)
4. Gemmell, J., Bell, G., Lueder, R.: MyLifeBits: A Personal Database for Everything. Communications of the ACM 49(1), 88–95 (2006)
5. Kim, K.-J., Jung, M.-C., Cho, S.-B.: KeyGraph-based Chance Discovery for Mobile Contents Management System. Knowledge-based and Intelligent Engineering Systems (KES Journal) (to appear, 2007)
6. Cho, S.-B, Kim, K.-J., Hwang, K.-S., Song, I.-J.: AnyDiary: Daily Cartoon-Style Diary Exploits Bayesian Networks. IEEE Pervasive Computing 6(3), 66–75 (2007)
7. Aizawa, K., Tancharoen, D., Kawasaki, S., Yamasaki, T.: Efficient Retrieval of Life Log based on Context and Content. In: Proceedings of the 1st ACM workshop on Continuous Archival and Retrieval of Personal Experiences, New York, USA, pp. 22–31 (2004)
8. Okazaki, N., Ohsawa, Y.: Polaris: An Integrated GPS Log Miner for Chance Discovery. In: Proceedings of the Third International Workshop on Chance Discovery and Its Management, Crete, Greece, pp. 27–30 (2003)
9. Ohsawa, Y.: Keygraph as Risk Explorer in Earthquake-sequence. Journal of Contingencies and Crisis Management 10, 119–128 (2002)
10. Blighe, M., Le Borgne, H., O'Connor, N., Smeaton, A.F., Jones, G.: Exploiting Context Information to Aid Landmark Detection in SenseCam Images. Communications of the ACM 49(1), 88–95 (2006)
11. Ohsawa, Y., Benson, E.N., Yachida, M.: Keygraph: Automatic Indexing by Co-occurrence Graph based on Building Construction Metaphor. In: IEEE ADL 1998. Proc. Advanced Digital Library Conference, pp. 12–18 (1998)
12. Horvitz, E., Dumais, S., Koch, P.: Learning Predictive Models of Memory Landmarks. In: CogSci 2004. 26th Annual Meeting of the Cognitive Science Society, Chicago, pp. 1–6 (2004)
13. Abowd, D.G., Dey, K.A., Brown, J.P., Davies, N., Smith, M., Steggles, P.: Towards a Better Understanding of Context and Context-Awareness. In: Gellersen, H.-W. (ed.) HUC 1999. LNCS, vol. 1707, pp. 304–307. Springer, Heidelberg (1999)

Topological Tree Clustering of Social Network Search Results

Richard T. Freeman

Capgemini, FS Business Information Management
No. 1 Forge End, Woking, Surrey, GU21 6DB
United Kingdom
richard.freeman@capgemini.com
http://www.rfreeman.net

Abstract. In the information age, online collaboration and social networks are of increasing importance and quickly becoming an integral part of our lifestyle. In business, social networking can be a powerful tool to expand a customer network to which a company can sell products and services, or find new partners / employees in a more trustworthy and targeted manner. Identifying new friends or partners, on social networking websites, is usually done via a keyword search, browsing a directory of topics (e.g. interests, geography, or employer) or a chain of social ties (e.g. links to other friends on a user's profile). However there are limitations to these three approaches. Keyword search typically produces a list of ranked results, where traversing pages of ranked results can be tedious and time consuming to explore. A directory of groups / networks is generally created manually, requires significant ongoing maintenance and cannot keep up with rapid changes. Social chains require the initial users to specify metadata in their profile settings and again may no be up to date. In this paper we propose to use the topological tree method to dynamically identify similar groups based on metadata and content. The topological tree method is used to automatically organise social networking groups. The retrieved results, organised using an online version of the topological tree method, are discussed against to the returned results of a social network search. A discussion is made on the criterions of representing social relationships, and the advantages of presenting underlying topics and providing a clear view of the connections between topics. The topological tree has been found to be a superior representation and well suited for organising social networking content.

Keywords: Information retrieval, social networking website, social networks, Web 2.0, semantic web, search engine optimization, document clustering, self organizing maps, topological tree, neural networks, post retrieval clustering, taxonomy generation, enterprise content management, enterprise search, information management.

1 Introduction

The exploding growth of web content is leading to an information overload, in which the use of web search engines is becoming critical to finding and retrieving relevant

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 760–769, 2007.

content. Despite the numerous advances in information visualisation [1], the most popular way of presenting search results still remain ranked lists. In this format, the user generally never looks beyond the first three pages, after which they will rather refine their search query by adding more terms or refining the initial query [2]. On the Web the results returned from web search engines, have been widely studied and Search Engine Optimisation (redesigning a website to improve its web pages ranking) is still a thriving industry [3]. However on social network websites less investigation has been made due to the complexity of the social ties and groups.

Recent research has been focusing on searching social network websites [4], as these have emerged as hugely popular and fast attracting a growing number of users. Some of the site are specialised on different areas, e.g. business (LinkedIn[1]), school/work based (FaceBook[2]), photo sharing (Flickr[3]) or general topics (MySpace[4]). Users generally upload content (e.g. photographs, videos, documents), post comments (blogs, discussions, bookmarks) and biographical information (e.g. name, university attended, current employer), and can often network as friends with other users of groups. With the rapidly growing number of profiles, groups and links, searches on social networking websites are emerging as an important tool for users to for example to find friends, business partners or groups with similar interests. Such searches currently heavily rely on metadata assigned by the user, e.g. a user in Facebook[5] has to provide biographical and personal details for other users to be able to retrieve their profile. One common approach to categorising the user profiles has been to allow them to join communities. This is an important aspect of social networks as these communities help bind the users together. These communities, networks or groups can be based on common interests, activities, or current school/work location.

There are generally three types of ways for a new user to join a community. The first is via browsing a directory where the communities have been manually organised by topics. However this is generally done manually leading to issues around maintenance and subjective interpretation. The second is done by a set of social ties or via recommendations, but such links have to be created manually. The third can be performed via a keyword and/or metadata search. Although ranking mechanisms help order the group profiles in terms of their relevance to the users query, they do not provide any guide as to the overall themes described in the pages or their relationships. Combining the approaches of search and directories has been done for some time in web search engines. For example some efforts have been made to provide different visual representation of the search results, such as suggesting keywords to refine the search (e.g. Webcrawler[6]), representing a graph view of the relations between pages (e.g. Kartoo[7]) or clustering the results (Vivisimo[8]). A major review of the methods and algorithms can be found in [5][6].

This paper deals with methods that organise groups (retrieved by a social network search engine) into a set of virtual folders which are labelled automatically using

[1] http://www.linkedin.com/
[2] http://www.facebook.com/
[3] http://www.flickr.com/
[4] http://www.myspace.com/
[5] http://www.facebook.com/
[6] http://www.webcrawler.com/
[7] http://www.kartoo.com/
[8] http://www.vivisimo.com/

extracted keywords. A method which clusters group pages dynamically, whilst creating a topology between them in a tree view, is presented in this paper. The topological tree method, first introduced by the author [5], is enhanced through weighting terms depending on their relation to the query term and making the algorithm function efficiently with dynamic social network datasets. Results and discussions confirm that the topological tree representation can be used to provide a user with a more intuitive and natural representation for browsing groups and discovering their underlying topics.

2 Visual Representation of Retrieved Social Content

One of the major growth and success factors of social network websites is that the users are repeatability returning to same website like in a real world community. One of the key factors for this being that the content is regularly updated. The content is a key driver that can be accessed through the networks, groups and user profiles. The access to such social content is typically performed using a search query, browsing through a directory, of though the social ties as shown in Table 1.

Table 1. Comparison of the differing methods for accessing the social content

Name	Description	Issues
Keyword / metadata search	Searches can be made on the name but also the content labels (e.g. photographs, videos, documents), posted comments (blogs, discussions, bookmarks), biographical information (e.g. interests, hobbies, university attended, current employer, geography) and groups (e.g. the groups the users decided to join).	The search is heavily reliant on manual tagging / labelling and the search results are usually presented in a ranked list.
Browsing the group, network or community directory	Typical social networking systems allow a user to visualise and browse a network directory of potential new friendships based on shared interests (e.g. creating new business contacts or finding romantic relationships).	The taxonomy directory might not be adequate for categorising all groups / networks, has to be manually maintained and can rapidly become out of date.
Chain of social ties	A chain of social ties can be used to link friends of friends, e.g. a user can discover friends in common through looking at such social ties. Previous work has suggested that social ties which link two individual are between five and seven [9].	The links are added manually and the search process can be time consuming and might have to be repeated if new links are formed or new profiles are added.

2.1 Searching for Profiles and Groups in a Social Network

A number of papers have been published on searching around the Semantic Web which has strong ties with social network searching. For example one author

suggested using semantic web analysis in the analysis of social networking website [10]. Community analysis helps finds community structure in social networks. For example the CNM algorithm, is a bottom-up greedy agglomerative clustering which selects and merges pairs of clusters by trying to maximise modularity of the community structure [11]. A more scalable community analysis algorithm has also been proposed [12].

Communities in Orkut (now owned by Google) have previously been analysed through the use of different similarity measure [13]. The main objectives were to evaluate different community similarity measure and recommended a ranked list of related communities, relative to a base community, that might be of interest to some users. The paper found that *L-2 Norm* (also known as cosine distance) showed the best empirical results. Another important finding was the impact of community size on the similarity measures, e.g. Mutual Information favours very large communities, while *L-1 Norm* favours small communities.

2.2 The Importance of Clustering and Topology

In information access systems, the major visual representations are Self-Organising Maps (SOMs), binary trees, balanced or unbalanced trees, graphs, and ranked lists. In some cases a combination of these representations can be used. This section describes the limitations of these methods, and illustrates the benefits of using the topological tree structure.

Clustering algorithms can be used to sort content into categories which are discovered automatically based on a similarity criterion. Its typical output representation is a binary tree or hierarchy. Binary trees quickly become too deep as each level only has two nodes; this representation has been used for retrieval rather than browsing. Hierarchies are typically generated using divisive partitioning algorithms (e.g. divisive *k*-means), or manually constructed such as with social bookmarks / folksonomies (e.g. Del.icio.us[9]) and web directories (e.g. Dmoz[10]). Web directories are particularly beneficial to users who are not familiar with the topics and their relations. However, even if some show cross links with related topics, they do not show the relations between topics at the same level, rather the topics are sorted alphabetically or by popularity. Other search engines such as Vivisimo do cluster results, however at each level in the tree there is always a category "other topics" where many document are clustered to. In addition, as with the other unbalanced trees, there is no relationship between the topics at each level.

Graph representations or SOMs can be used to compensate for this lack of topology in these tree representations or taxonomies. Graphs can represent hyperlinks, relationships or links between topics. A web example of a graph generated representation is Kartoo. Other knowledge representations such as Topic Maps (e.g. Omnigator[11]), can also be represented as graph structures. Although they do capture the inter topic / document relations, the major drawback is that they cannot scale easily, i.e. the more nodes / links are added the less legible it becomes. SOMs typically have a 2-dimensional grid structure which adapts to the content space and

[9] http://del.icio.us/
[10] http://www.dmoz.org/
[11] http://www.ontopia.net/omnigator/models/index.jsp

the number of nodes need not change to represent the underlying number of topics. The SOM-based methods have two distinct properties over other methods, namely non-linear dimensionality reduction and topology preservation. The non-linear projection property ensures that the input space is mapped onto a lower dimensional space with minimum information distortion. The topology preserving clustering enables documents that are similar to be located closely on the map. However one the major weakness of 2-dimensional SOMs, is it is difficult to navigate between different levels of detail. Hierarchical variants of the SOM, such as the Growing Hierarchical SOM [7] have been developed for this purpose; however only one map can be shown at any time and their size is sensitive to fixed parameters. In addition, tables or complex graphics are required to represent the 2-dimensional maps efficiently.

The topological tree method, first proposed by the author [5], compensates for all these factors by exploiting a simple tree view structure to represent both *hierarchical* and *topological relationships* between topics. Previous work undertaken by the author focused on clustering a fixed set of documents. This paper deals with the clustering of search results of multi author / non-uniform documents with different formatting and content. The topological tree can be used to combine the tree structure typically used in file explorers with that of the topology inherent in SOMs. The tree structure allows a user to visualise different levels of detail and hierarchical relationships. The topology, a novel feature specific to the topological trees and SOMs, additionally allows the viewing of the relationships between the topics. Fig. 1 clearly shows the difference between having a topology and not having one. On the left, the topics appear to be randomly placed, but on the right they seem to naturally flow downward as economics, microeconomics, finance, biology, and anatomy making it more intuitive and natural to the user.

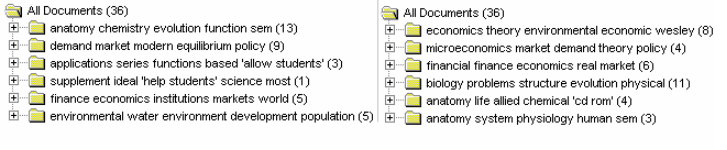


Fig. 1. – *k*-mean with no topology (left) and root level in the topological tree (right)

3 The Topological Tree Method

3.1 Overview of the Method

There are a number of essential steps in the method:

1. The user enters a query term into the local web application, and selects the search options and social network search engine.
2. The application submits the query term to the social search engine and crawls the returned results.
3. Each page is indexed and transformed into a document vector.
4. Feature selection and term weighting is performed on the vector.
5. The documents are organised in a growing chain.

6. Each chain is labelled and added to the topological tree, if further child chains are required return to 4.
7. The user is presented with the resulting generated topological tree.

3.2 Growing Chains and Topological Tree Method

SOMs are generally associated with 2-dimensional structures that help visualise clusters and their relationships in a topology. However, equally 1-dimensional chains can also be used. The topological tree method uses 1-dimensional chains where each node may spawn a child chain. The number of nodes in each chain is guided by an independent validation criterion. The algorithm used to grow the 1-dimensional SOM is termed growing chain (GC) and shares growing properties with the growing grid (used in the GH-SOM [7]) and growing SOM variants, but is more suited for 1-dimension.

In a similar way to the SOM, there are two major steps in the GC algorithm: the search for the best matching unit and the update of the winner and its neighbouring nodes. At time t, an input document vector $\mathbf{x}$ is mapped to a chain consisting of n nodes with a weight vector $\mathbf{w}$. The best matching unit $c(\mathbf{x})$ is the node with the maximum dot product amongst nodes j and document vector $\mathbf{x}(t)$,

$$c(\mathbf{x}) = \arg\max_j \left\{ S_{dot}(\mathbf{x}(t), \mathbf{w}_j) \right\}, \quad j = 1, 2, ...n \tag{1}$$

where n is the current number of nodes. Once the winner node $c(\mathbf{x})$ is found the neighbouring weights are updated using,

$$\mathbf{w}_j(t+1) = \frac{\mathbf{w}_j(t) + \alpha(t) h_{j,c(x)}(t) \mathbf{x}(t)}{\left\| \mathbf{w}_j(t) + \alpha(t) h_{j,c(x)}(t) \mathbf{x}(t) \right\|} \tag{2}$$

where $\alpha(t)$ is the monotonically decreasing learning rate and $h_{j,c(x)}(t)$ the neighbourhood function, typically a Gaussian kernel. When the learning has stabilised for the current number of nodes n, the entropy of the chain is recorded and a new node is inserted next to the node with the highest number of wins. The weights of the new node are initialised by interpolating or extrapolating existing nodes weight values. New nodes are added until n_{max} nodes are reached which corresponds to the maximum allowable chain size. Finally the validation criterion, the entropy-based Bayesian Information Criterion that penalises complexity, gives the optimum number of nodes per chain as:

$$\tau = \arg\min_n \left\{ \frac{1}{m} \sum_{j=1}^{n} m_j \cdot H(C_j) + \frac{1}{2} n \log m \right\}, n = 2, .., n_{\max} \tag{3}$$

where m is the number of documents, n the current number of nodes in the chain, $H(C_j)$ is the total normalised and weighted sum of entropies for cluster C_j.

Then in the hierarchical expansion process, each node in the chain is tested to see if it will spawn a child chain. This is performed using several tests. The first test counts the number of document clustered to that node to see if it is less than a fixed threshold. The next test analyses the vocabulary present in those documents to

determine if there is a sufficient number of terms. The final test uses cluster tendency method. It aims to test if a set of documents contains random documents with no or few relations or if there are strong underlying clusters [8]. If any of these tests fail for a particular node, then it does not spawn a child chain and becomes a leaf node in the final topological tree representation.

Finally each node in the chain is labelled using the most representative terms of the node's weight and its frequency. Once the chain is labelled, then it is added to the current topological tree structure. If further hierarchical expansions in its child chains are required, then the process is repeated for each of the child chains, otherwise the process is terminated and the results presented to the user. The full pre-processing and topological tree method is shown in Fig. 2.

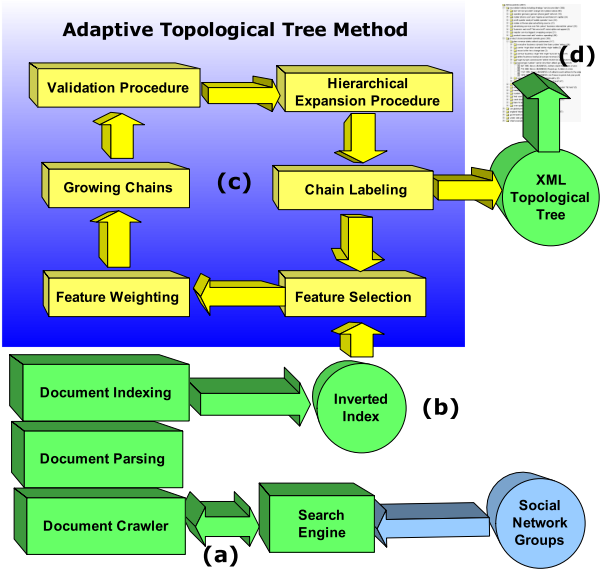


Fig. 2. The Topological Tree Method. (a) The search engine is queried, the pages are returned and crawled by the Web Application. (b) An inverted index is generated from the retrieved documents. (c) The closed loop represents the necessary processing for each growing chain in the topological tree. It is grown using an independent validation procedure that estimates the optimum number of nodes that maximise the information value. (d) Once the topological tree is complete it is exported to XML.

4 Results and Discussions

The dataset was dynamically generated from a search query for group pages in MySpace. The query was "developer"; other queries were also tested but omitted for space considerations. The MySpace tree, shown in Fig. 3, was generated by directly

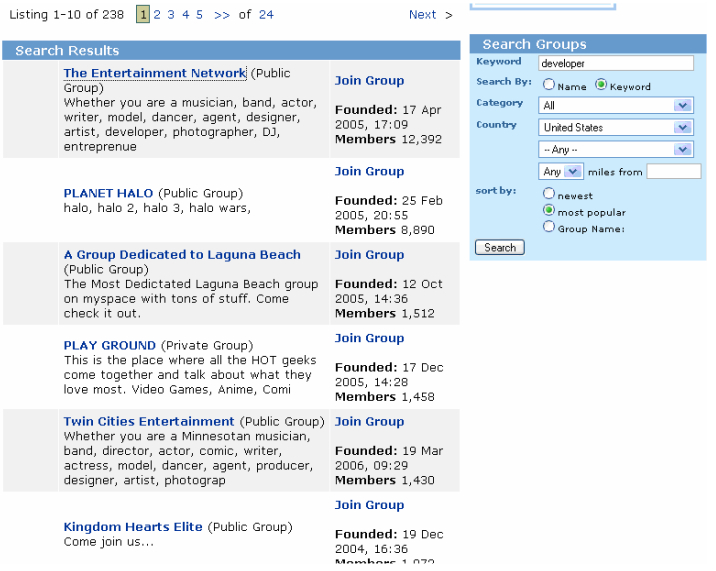


Fig. 3. – A partial snapshot of a search results for groups returned on the query "developer" in MySpace Social networking website. Clearly the sorting by newest, most popular (i.e. size) or group name (alphabetical) is not the best way of organising groups. The user cannot gain an insight of the types of groups without browsing though several pages of the search results or identify the relationship between the groups.

submitting the same query to the search engine and taking a sample snapshot of the results. Fig. 4. shows the topological tree that was generated from running a query and crawling the returned ranked listing.

4.1 Comparison and Discussion

MySpace currently uses simple search criterion of keyword, category and country for retrieving groups. The main limitations of these are that this relies on correct tagging and updating of categories listing. The results can be sorted by newest, most popular (i.e. size) or group name (alphabetical), however a user will still have to browse through pages of results to identify different types of groups they might be interested in. In comparison, the topological tree representation appears more intuitive and natural to the user, as closely related topics are located close to one another in each chain. Each chain does not grow to a large number of nodes, as this number is guided by an independent validation criterion that penalises complexity. In addition hierarchical relations between a parent node and child chain help abstract different levels of detail.

There are four important criteria for creating an effective browsing experience of documents and topics:

1. *Hierarchical Representation*: the topics need to show different levels of detail simultaneously. This is especially true when the number of topics is large, e.g. the Dewey decimal classification or web directories.

2. *Scalability*: the ability to view a large number of topics and documents in the same window.
3. *Visualise key topics and their related documents*: key topics should be easily be discernable using a label and documents should be shown to belong to one of more of them.
4. *Visualise key relationships*: the ability to visualise the relationships between different topics as well as the connections between documents.

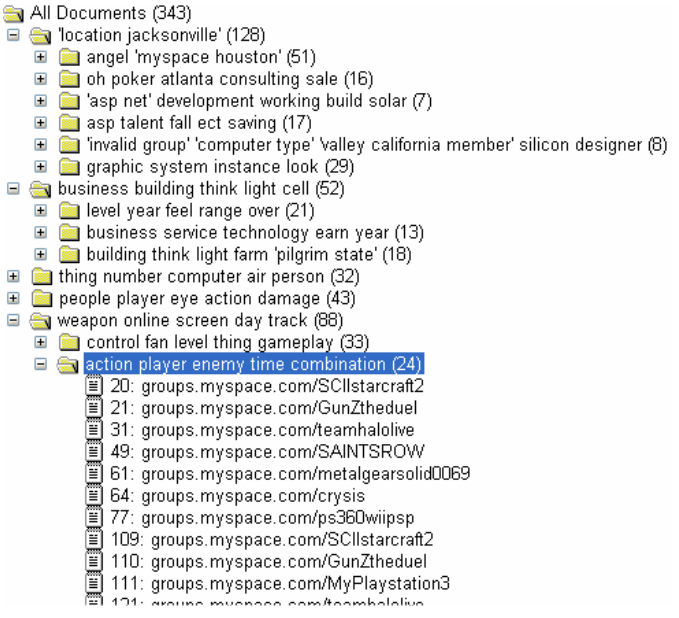


Fig. 4. – A topological tree generated from the pages retrieved using the query "developer" (for all countries). Clearly the topology ensures that the property developer, software developers and games developers are closely grouped together in a more intuitive way.

5 Conclusion and Future Work

A topological tree is a tree view structure that does not require complex 2-dimensional graphics or tables such as used in SOMs or graphs. Yet it can show the key relationships between extracted topics thus helping reveal previously unknown associations automatically. It also helps make a tree structure appear more intuitive, i.e. related topics are located close to one another in the tree. This topology can be thought of as a graph representation that has been optimised into a tree view, where only the strongest relationships between topics are preserved. Through building on top of existing search engines, the topological tree method benefits from pre-filtered content where it only needs to organise a relevant subset of the content. This paper has shown that the topological tree can be built on top of a typical social search engine and produce an insightful overview of the underlying topics contained in the

top ranking MySpace groups. Future work could look at extracting and combining knowledge from web directories and social networks, with results returned from a web search engine, into a topological tree.

References

[1] Herman, I., Melancon, G., Marshall, M.: Graph visualization and navigation in information visualization: A survey. IEEE Transactions on Visualization and Computer Graphics 6(1), 24–43 (2000)
[2] Search Engine User Behavior Study, White Paper, iProspect (2006)
[3] Zhang, J., Dimitroff, A.: The impact of webpage content characteristics on webpage visibility in search engine results (Part I). Information Processing and Management 41, 665–690 (2005)
[4] Adamic, L., Adar, E.: How to search a social network. Social Networks 27(3), 187–203 (2005)
[5] Freeman, R.T., Yin, H.: Adaptive topological tree structure for document organisation and visualisation. Neural Networks. 17(8-9), 1255–1271 (2004)
[6] Freeman, R.T.: Web Document Search, Organisation and Exploration Using Self-Organising Neural Networks, PhD Thesis, Faculty of Engineering and Physical Sciences, School of Electrical & Electronic Engineering, University of Manchester: Manchester (2004)
[7] Rauber, A., Merkl, D., Dittenbach, M.: The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. IEEE Transactions on Neural Networks 13(6), 1331–1341 (2002)
[8] Freeman, R.T., Yin, H.: Web content management by self-organization. IEEE Transactions on Neural Networks 16(5), 1256–1268 (2005)
[9] Dodds, P.S., Muhamad, R., Watts D.J.: An Experimental Study of Search in Global Social Networks. Science 301 (2003), www.sciencemag.org
[10] Mika, P.: Flink: Semantic Web Technology for the Extraction and Analysis of Social Networks 3(2-3), 211–223 (2005)
[11] Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E70, 66111 (2004)
[12] Wakita, K., Tsurumi, T.: Finding Community Structure in Mega-scale Social Networks. In: WWW 2007 (May 8-12, 2007)
[13] Spertus, E., Sahami, M., Buyukkokten, O.: Evaluating Similarity Measures: A Large Scale Study in the Orkut Social Network. In: KDD 2005, pp. 678–684 (August 21-24, 2005)

A Framework to Analyze Biclustering Results on Microarray Experiments

Rodrigo Santamaría, Roberto Therón, and Luis Quintales

University of Salamanca

Abstract. Microarray technology produces large amounts of information to be manipulated by analysis methods, such as biclustering algorithms, to extract new knowledge. All-purpose multivariate data visualization tools are usually not enough for studying microarray experiments. Additionally, clustering tools do not provide means of simultaneous visualization of all the biclusters obtained.

We present an interactive tool that integrates traditional visualization techniques with others related to bioinformatics, such as transcription regulatory networks and microarray heatmaps, to provide enhanced understanding of the biclustering results. Our aim is to gain insight about the structure of biological data and the behavior of different biclustering algorithms.

1 Introduction

Biclustering methods are techniques that discover internal structure of data in a non-supervised way. In the last few years they have been extensively applied to bioinformatics, specially to extract knowledge from microarray experiments. The first effort was done by Cheng and Church [5]; many others are surveyed in [13,23]. Nowadays, still new biclustering methods are developed [15,4].

On the other hand, there are tools covering different aspects of biological and statistical analysis. BicAT [2] is a great tool focused in biclustering algorithms, implementing some of the most important ones along with traditional k-means and hierarchical clustering. BicAT presents the results as ordered lists of biclusters, that can be examined individually through heatmaps and parallel coordinates.

Expander [20,18] is also a tool that implements clustering and biclustering methods. Although Expander implements less biclustering algorithms than BicAT, it has a great number of visualizations: heatmaps and boxplots to study microarray data matrices, dendrogram+heatmap visualization of hierarchical clustering results [7], clustering PCA displays and bicluster heatmaps. The PCA display may be the most interesting view because it allows a quick understanding of gene structure (coloring points depending on the cluster which groups them).

gCluto [16] makes use of more advanced information visualization techniques. The microarray data matrix is again represented by a heatmap but now the interaction with the representation is allowed, so rows and columns can be expanded, combined or grouped by hierarchical clustering. gCluto also uses 2D projections of clusters but in a 3D space called mountain maps, where perimeter, height, slope and color identify different properties of each cluster.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 770–779, 2007.

The Rank-by-feature framework [17] is another powerful tool for hierarchical and k-means clustering. In this case a great level of interaction is allowed, under a high number of views: heatmaps, dendrograms, histograms, scatter plots and parallel coordinates. Finally, Cytoscape [19] is a very different tool, focused in analyzing biomolecular interaction networks with an optimal degree of interaction (zooming, searching, changes of layout, coloring, database querying and lots more).

Although the aforementioned tools deal with clustering and/or biclustering results, they do not focus on the simultaneous visualization of them. BicAT visualizes biclustering results individually, and comparison must be done through navigation of lists, which makes difficult the discovery of relationships among biclusters. Expander and gCluto present different solutions to this but for clustering results. The representation of multiple biclustering results of one or more biclustering methods has not been treated.

To overcome these limitations, we have developed a visual analysis tool that allows the simultaneous display of all the biclustering results of different methods along with linked views of related information, such as microarray expression levels and transcription regulatory networks (TRNs). That way, a full framework to help in decision making has been implemented and tested.

The following sections are organized as follows. Section 2 exposes the visualization techniques implemented in the tool: definition of the structure, data, displays, user interactions implemented and linkages between views. Section 3 presents a full example of the use of the framework with a synthetic microarray data experiment. Finally, Section 4 draws the conclusions achieved and establishes future lines for expanding the tool.

2 Bicluster Visualization

The framework manages different data sources and display them by using a number of visualizations techniques. All the visualizations are interconnected by means of a session manager to allow flow of data and interactions among views (see fig. 1). Three data sources are distinguished. The most important is the Microarray Data Matrix, that contains information about gene names, condition details and gene expression levels. Following, TRN network, represented as an XML standard graph, provides information about genes and relationships between them (up or down-regulation). Finally, bicluster results are presented as an structured file with information about the type of biclustering algorithm, the dimension of the biclusters and the genes and conditions grouped by them.

These data are visualized by means of five main visualization techniques: heatmaps, parallel coordinates, scatter plots, bubble maps and transcription graphs (Fig. 2). The first three visualizations represent microarray expression levels as multivariate data where each gene or sample is a variable and each condition or experiment is a dimension. The tool also allows the presentation of this data as a textual table. The bubble map represents biclustering results while the transcription graph represents a TRN of the organism studied in the microarray. For description purposes, we will use *gene* to address to a variable and *condition* for dimensions. We will have n genes $G = \{g_1, ..., g_n\}$ and m conditions $C = \{c_1, ..., cm\}$. A bicluster B is a subset of n_b genes ($G_b = \{g'_1, ..., g'_{nb}\}$) and m_b conditions ($C_b = \{c'_1, ..., c'_{mb}\}$).

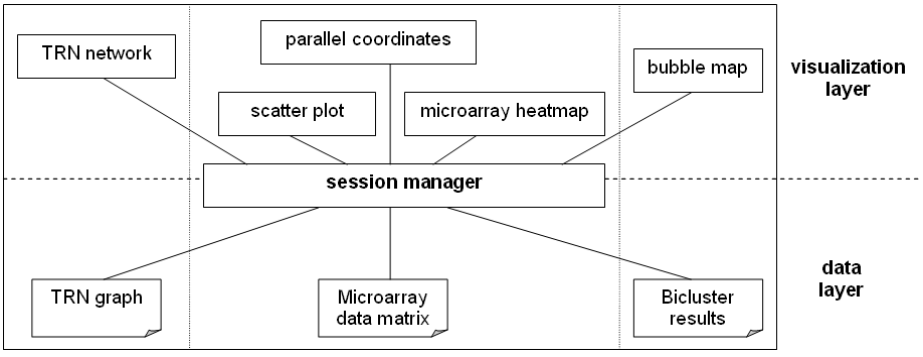


Fig. 1. Diagram of the structure of the framework. Three data sources can be used in the visualization of different displays by means of a session manager that interconnects them all.

2.1 Microarray Data Visualizations

Heatmaps (Fig. 2c) are the most usual representation of microarray data. In order to inspect genes or conditions individually, the heatmap implements bifocal distortion [12] by rows and/or columns, as well as zoom and navigation through expression levels. Selection of rows, columns or individual expression levels are linked to the other visualizations of the framework.

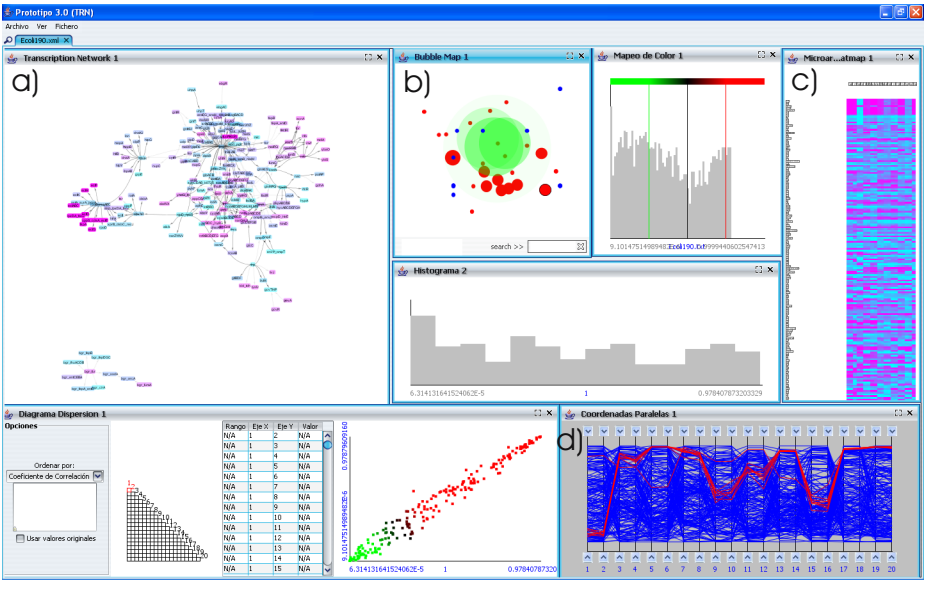


Fig. 2. Overview of the framework. Data belongs to the example discussed in Section 3. The most relevant visualizations are (a) TRN network, (b) bubble map, (c) microarray heatmap and (d) parallel coordinates.

Parallel coordinates (fig. 2d) represent G as a set of lines of m-dimensional points. Selection of ranges of values on any condition can be done. Conditions also can be reordered as desired.

2.2 Bubble Map

Bubble maps (fig. 2b) are related to gCluto mountain maps, but unlike gCluto maps, this visualization makes use of two dimensions to avoid 3D overlapping and improve time performance, allowing simultaneous comparison of a large number of biclustering results from different methods.

Each bicluster B is represented as a circle (bubble), where color identifies the biclustering method that computed it. The radius of the shape refers to the size of the bicluster, computed as $n_b m_b$. The transparency depends on bicluster homogeneity, defined as the inverse of the within variation described in eq. 1:

$$W_b = \frac{1}{n_b} \sum_{i=1}^{n_b} \sqrt{\sum_{j=1}^{m_b} (\bar{a}_j - a_{ij})^2} \tag{1}$$

where a_{ij} is the expression level of the gene g_i under the condition c_j and $\bar{a}_j$ is the mean of the expression levels of the genes grouped in B for condition j.

The position is determined by the genes and conditions grouped. The horizontal coordinate depends on conditions while the vertical coordinate depends on genes. To compute the positions, bicluster B, grouping gene subset G_b and condition subset C_b, is mapped to the multidimensional points x_b and y_b as in eqs. 2 and 3.

$$x_b = (p_1, p_2, ..., p_n) \mid p_i = 1 \Leftrightarrow g_i \subset G_k,\ p_i = 0\ otherwise \tag{2}$$

$$y_b = (p_1, p_2, ..., p_m) \mid p_j = 1 \Leftrightarrow c_j \subset G_k,\ p_i = 0\ otherwise \tag{3}$$

These two points of n and m coordinates are projected to one dimension with either a classical metric [8] or non-metric [10] multidimensional scaling. This way both y-axis and x-axis components of the representation for each bicluster are obtained. Therefore, biclusters at the same horizontal/vertical line are expected to share genes/conditions, although this is not always precise due to the reduction of dimensionality, that obviously loses information.

The result is a set of distributed, colored, sometimes superposed circular shapes, where an analyst can easily identify biclusters distant from the trend, differences between biclustering methods or other relevant knowledge (Figs. 2b, 3a). The user can select any number of biclusters, a change that is transferred to other views to highlight the corresponding genes and/or conditions. Bubbles can be dragged to change their positions in case the user wants to reorder them using any other criterium.

2.3 TRN Visualization

In a TRN, nodes represent the set of all genes G, while a directed edge from g_i to g_j means that g_i encodes for a transcription factor protein that transcriptionally regulates

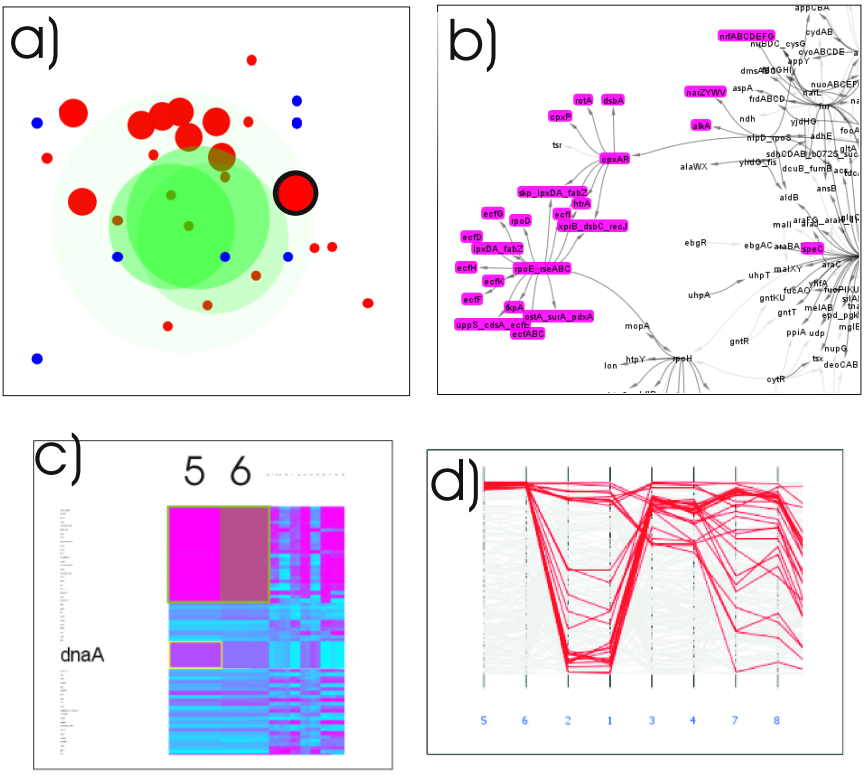


Fig. 3. Example of how linkage works. A Bimax (red) bicluster is selected in bubble map (a) and this provokes gene highlighting in TRN (b), reordering of rows and columns in heatmap (c) and highlighting of lines and reordering of axis in parallel coordinates (d). Similar flows can be followed by interaction with other visualizations.

g_j [14]. It is important to distinguish at least two types of edges: activation and repression edges. When a gene up-regulated connects with an activation edge to another gene, this one is favored to up-regulation. If it connects with a repression edge, will be favored to down-regulation.

In our framework, TRNs have been represented as directed acyclic graphs led by forces (Fig. 2a). Nodes are labeled with gene names and edges are colored in dark or light grey depending if the interaction is activation or inhibition, respectively. To avoid edge cluttering, they are displayed with splines instead of straight lines. We also implement a gene search by name. The interacting forces display the nodes so the overlapping of nodes and edges is minimized.

2.4 Linked Visualizations

All the visualizations are linked so changes in a view are propagated to the rest of views (Fig. 3). The ability of visualizing changes in a representation because of interaction

with another representation helps to reveal patterns. On the other hand, linkage limits the screen area because it has to be divided by different visualizations. All linkages implemented are bidirectional, so flow between visualizations can be followed at user's demand.

In our case, the usual flow of information that communicates views are subsets of genes and/or conditions. Thus, a selection of a node in the TRN will imply the flow of the gene represented by that node to other views, highlighting biclusters that contain this gene or focusing on the gene in the microarray heatmap, for example. The user can configure which visualizations to monitor simultaneously and if they are linked or not, thus adapting screen areas to her necessity.

3 Case Study

3.1 Example Dataset

In order to make the discussion simpler, we have chosen a reduced synthetic example obtained by SynTReN [6] from Shen-Orr's E. coli TRN [21]. From this network, with 424 nodes, SynTReN builds a synthetic TRN with 200 nodes, 190 nodes based in Shen-Orr's definition and 10 random nodes, without biological basis. SynTReN will also generate a microarray data matrix simulating 10 experiments, each one repeated two times.

We apply three different biclustering algorithms to the microarray data matrix: Bimax [15], Plaid models [11] and Spectral biclustering [9]. We have chosen methods that differ in its interpretation of biclusters, so it is expected that their results will be quite distinct.

3.2 Objectives

The framework has been designed in such a way that analysis will naturally follow the Information Visualization Mantra: "Overview first, zoom and filter, details on demand" [22]. This way, it will start with a general overview of our problem, to continue with filterings by biclusters, genes and conditions. With these flows, supported by linked views, we will prove the potence of the framework to analyze the mentioned dataset regarding the following: 1) detecting relationships between the two replications of each experiment, 2) determining characteristics in the biclusters computed by different methods, 3) checking if related groups in TRN are grouped by biclusters and 4) detecting random genes, and determining if they appear in the biclusters computed.

Additionally, we want to discover: 1) new relationships between genes not related in the TRN, 2) biclusters deviated from the trend and 3) differences and similarities of the three biclustering methods and its performance for this example.

3.3 Overview

A simple overview using different visualizations gives interesting information. The TRN layout (fig. 2a) shows how genes are related according to existing biological knowledge. A group of random genes is easily detected as a separate graph at the

bottom-left. The bubble map (fig. 2b, 3a) shows biclusters for Bimax (red), Plaid model (green) and Spectral (blue). With just a glance, we can tell than Plaid model gives bigger, heterogeneous (transparent) biclusters (due to some extent by a reported problem of this algorithm [24]), while Spectral biclustering gives very small ones and are displayed linearly, revealing the checkerboard structure of Spectral biclustering. Bimax returns middle-size, homogeneous (solid) biclusters. Also, biclusters deviated from the trends and groups of neighbor biclusters are easily detected, possibly worth a deeper study with the tool. The microarray heatmap and parallel coordinates are not very helpful on an overview, being the expression level information overwhelming without previous filtering. Finally, a scatter plot comparing expression levels of different replications of the same experiment (fig 2, bottom left) reveals its correlation.

3.4 Bicluster-Oriented Analysis

Once the overview has given us a context to draw preliminary analysis, deeper exploration is needed. This usually starts with biclusters, displayed with different colors depending on their method of biclustering. Interesting biclusters because of their homogeneity, size or position are salient in the bubble map visualization and can be selected, provoking changes in other visualizations that give us insight about what is grouped in the bicluster and why.

The microarray heatmap will reorder and highlight genes and conditions on the bicluster, giving a quick way to identify what is in the bicluster. Also heatmaps, along with parallel coordinates, help to understand why these genes and conditions are grouped together by the algorithm in terms of their expression levels. For example, when selecting a Bimax bicluster as in fig. 3, genes highlighted in heatmap and parallel coordinates present high and constant expression levels through the corresponding conditions. These are two of the features of Bimax algorithm, and therefore the information helps us to confirm that the results are correct or (if the biclustering method is not well known) to learn about the biclustering behavior. On the other hand, when a bicluster is selected, the corresponding genes highlight in the TRN network. Usually, as in fig. 3b, groupings are reflected in previously biological relationships (left bunch of genes) but in some cases previously unrelated genes are grouped, as it is the case of the gene at the right of the figure. Thanks to the force layout of the TRN graph, genes unrelated (very separated) can be easily detected.

Various biclusters can be selected simultaneously, thus highlighting in other visualizations the intersecting genes and conditions. This is interesting when clouds of biclusters are detected in the bubble map.

3.5 Gene and Condition-Oriented Analysis

Studying the biclusters, some genes appear grouped without direct (or obvious indirect) relation in the TRN. These genes could be actually related or be misgrouped by biclustering algorithms. If that kind of genes are grouped by a large number of biclusters, the probability of them being really related increases, justifying further analysis. The same is valid with conditions.

To analyze these interesting objects, we can change the scope and flow of the navigation through the tool and start by selecting particular genes. Picking those genes in

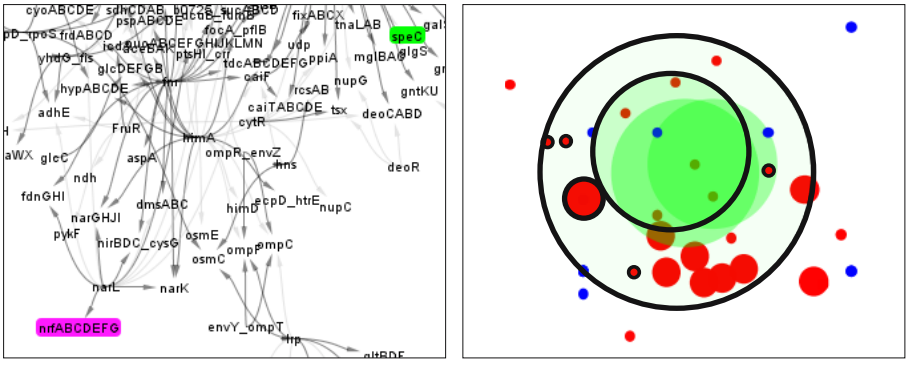


Fig. 4. Genes named *spec* and *nrfABCDEFG* are grouped together in seven biclusters from two different biclustering methods, without known biological evidence. The framework helps to discover it quickly.

the TRN will highlight all the biclusters that groups them together. If a high number of biclusters is highlighted, it is possible that the genes are truly related and we have discovered relevant knowledge (Fig. 4).

4 Conclusion and Future Work

A framework to study biclustering methods in terms of its results by different visualizations, including biological knowledge with TRNs, is presented. The use of this framework, along with benchmark datasets and statistical and biological validation techniques can shed more light on performance of biclustering methods. It also will help analysts in the study of the usually large number of biclusters given by biclustering algorithms, decreasing analysis time and helping in the detection of relevant results. The tool discussed has relevant advantages over other current tools:

- Visualization of all biclusters *simultaneously* by means of the bubble map. This visualization also allows the representation of biclusters from different biclustering algorithms simultaneously. Only gCluto and Expander implements simultaneous visualization of simple clusters form a single method, without interaction.
- Incorporation of biological information from transcription regulatory networks to the visualization of microarray data and biclusters, allowing their communication. This is an unusual feature, only implemented by Expander (by means of visualization of transcription binding sites in gene sequences) and Cytoscape (coloring of TRNs by expression levels).
- Simultaneous visualization and linking between different views. This is a key concept to increase the user's insight on the problem, witnessing the changes that interaction with a visualization causes in other views.
- Use of statistical measures such as coherence and variance by means of bubble map, thus including another relevant aspect of biclustering analysis: validation metrics.

Aside for the aforementioned advantages, new paths to improve the tool are opened:

- The bubble map, although useful, is based in projections that reduce dimensionality at the cost of discarding details. The result is that the overlapping of bubbles does not exactly convey the real overlapping of biclusters. Another technique is being currently studied to solve this.
- More biological knowledge will be, specially network motifs [14] identified in TRNs and GO [1] and MIAME annotations [3], increasing the details-on-demand.
- Gene and bicluster-oriented analysis discussed here are just two ways of revealing new knowledge. Testing of the tool by analysts will reveal new requirements in both visualization and genomic/transcriptomic areas.

Acknowledgements

The authors wish to thank Javier Molpeceres its contributions to programming this tool. This work was supported by the Education and Science Ministry of Spain under project TIN2006-06313 and by a grant from the Junta of Castilla y León.

References

1. Ashburner, M., Ball, C.A., Blake, J.A., Bolsteing, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Geneontology: tool for the unification of biology. Nature Genetics 25, 25–29 (2000)
2. Barkow, S., Bleuer, S., Prelic, A., Zimmermann, P., Zitzler, E.: Bicat: a biclustering analysis toolbox. Bioinformatics 22(10), 1282–1283 (2006)
3. Brazma, A., Hingamp, P., Quackenbush, J., Sherlock, G., Spellman, P., Stoeckert, C., Aach, J., Ansorge, W., Ball, C., Causton, H., Gaasterland, T., Glenisson, P., Holstege, F., Kim, I., Markowitz, V., Matese, J., Parkinson, H., Robinson, A., Sarkans, U., Schulze-Kremer, S., Stewart, J., Taylor, R., Vilo, J., Vingron, M.: Minimum information about a microarray experiment (miame)-toward standards for microarray data. Nat. Genet. 29(4), 365–371 (2001)
4. Carmona-Saez, P., Pascual-Marqui, R.D., Tirado, F., Carazo, J.M., Pascual-Montano, A.: Biclustering of gene expression data by non-smooth non-negative matrix factorization. BMC Bioinformatics 7(78) (2006)
5. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Proc. Int'l Conf Intell Syst Mol Biol., vol. 8, pp. 93–103 (2000)
6. den Bulcke, T.V., Leemput, K.V., Naudts, B., van Remortel, P., Ma, H., Verschoren, A., Moor, B.D., Marchal, K.: Syntren: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. BMC Bioinformatics 7(43) (2006)
7. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. 95, 14863–14868 (1998)
8. Gower, J.C.: Some distance properties of latent root and vector methods used in multivariate analysis. Biometrika 53(3-4), 325–338 (1966)
9. Kluger, Y., Basri, R., Chang, J.T., Gerstein, M.: Spectral biclustering of microarray data: Coclustering genes and conditions. Genome Research 13, 703–716 (2003)
10. Kruskal, J.B.: Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika 29(1), 1–27 (1964)

11. Lazzeroni, L., Owen, A.: Plaid models for gene expression data. Technical report, Standford University (2002)
12. Leung, Y.K., Apperley, M.D.: A review and taxonomy of distortion-oriented presentation techniques. Transactions of Computer-Human Interaction 1(2), 126–160 (1994)
13. Madeira, S., Oliveira, A.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Transactions of Computational Biology and Bioinformatics 1(1), 24–45 (2004)
14. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science 298, 824–827 (2002)
15. Prelic, A., Bleuer, S., Zimmermann, P., Wille, A., Bhlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9), 1122–1129 (2006)
16. Rasmussen, M., Karypis, G.: gcluto: An interactive clustering, visualization and analysis system. Technical Report 04-021, University of Minnesota (2004)
17. Seo, J., Shneiderman, B.: A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In: IEEE Symposium on Information Visualization, pp. 65–72 (2004)
18. Shamir, R., Maron-Katz, A., Tanay, A., Linhart, C., Steinfeld, I., Sharan, R., Shiloh, Y., Elkon, R.: Expander - an integrative program suite for microarray data analysis. BMC Bioinformatics 6(232), 1471–2105 (2005)
19. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T.: Cytoscape: A software environment for integrated models of biomolecular interaction networks. Genome Research 13, 24958–25004 (2003)
20. Sharan, R., Maron-Katz, A., Shamir, R.: Click and expander: a system for clustering and visualizing gene expression data. Bioinformatics 19(14), 1787–1799 (2003)
21. Shen-Orr, S.S., Milo, R., Mangan, S., Alon, U.: Network motifs in the transcriptional regulation network of escherichia coli. Nature Genetics 31, 64–68 (2002)
22. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: IEEE Visual Languages, number UMCP-CSD CS-TR-3665, pp. 336–343, College Park, Maryland 20742, U.S.A. (1996)
23. Tanay, A., Sharan, R., Shamir, R.: Biclustering algorithms: A survey. Handbook of Computational Molecular Biology (2004)
24. Turner, H., Bailey, T., Krzanowski, W.: Improved biclustering of microarray data demonstrated through systematic performance tests. Computational Statistics and Data Analysis 48, 235–254 (2003)

Methods to Bicluster Validation and Comparison in Microarray Data

Rodrigo Santamaría, Luis Quintales, and Roberto Therón

University of Salamanca

Abstract. There are lots of validation indexes and techniques to study clustering results. Biclustering algorithms have been applied in Systems Biology, principally in DNA Microarray analysis, for the last years, with great success. Nowadays, there is a big set of biclustering algorithms each one based in different concepts, but there are few intercomparisons that measure their performance. We review and present here some numerical measures, new and evolved from traditional clustering validation techniques, to allow comparisons and validation of biclustering algorithms.

1 Introduction

Biclustering is one of the main options to find structure in gene microarray data. In the last years, lots of biclustering methods have been proposed [10]. Authors apply different procedures to individually validate them. Also, with the growing number of algorithms, its comparison is now being addressed [12]. Though not an optimal algorithm exists, these comparisons help to understand biclustering behavior and make easier the choice of the bests algorithms in each context.

Several measures for validation exist in clustering area, but they are usually not applied for biclustering methods. The authors that have treated more in deepness comparison methodologies for biclustering are Prelic et al. [12] and Turner et al. [14]. Validation and comparison are made by external indices. Non-biological indices as sensitivity and specificity are used when information of clustering is known, usually in synthetic data where biclusters are embedded. Only constant and additive biclusters are treated, as they are the most extended. Biological indices are used when no information intrinsic to the data is known. Internal and relative indices are seldom used because biclustering concepts are hard to adapt to clustering indices.

In this paper, we review these validation and comparison techniques, explaining the adaptations done in literature and proposing some other adaptations to biclustering characteristics. Specially, internal and relative index application to optimize input parameters and coherence measures have been developed. In Section 2, we discuss the different kinds of biclusters offering measures to determine each type. Section 3 covers the use of internal, external and relative indices, reviewing the most used and extending some of them to biclustering context. Section 4 makes a brief application of measures discussed in Section 2 and 3 on two biclustering algorithms. Finally, Section 5 presents the conclusions and future work.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 780–789, 2007.

2 Bicluster Structure

2.1 Bicluster Classification

A bicluster can be defined as 'a subset of objects (rows or columns) that jointly respond across a subset of other objects (columns or rows)'. In bioinformatics, rows usually refer to genes and columns to experiments or organism conditions. Madeira and Oliveira [10] classify biclusters depending on what is considered for 'jointly responds':

- Constant value bicluster (C): all elements have exactly the same value (μ). Elements of constant bicluster $B = [b_{ij}]$ with n rows and m columns are defined as

$$b_{ij} = \mu \tag{1}$$

- Coherent value bicluster (H): row and/or column variations are somehow related. This relationship may be additive (H^+), multiplicative ($H^\times$) or by sign ($H^\pm$). In case of H^+ and $H^\times$, each row and/or column differs from others in an additive or multiplicative factor (eqs. 2 and 3, respectively). In case of $H^\pm$, it is just a qualitative rule of change in tendency (α and β are binary vectors representing increasing or decreasing respect to another row or column –such as 1 or -1–, but it's not imposed any quantitative restriction on r_{ij}, c_{ij} variations)

$$b_{ij} = \mu + \alpha_i + \beta_j \tag{2}$$

$$b_{ij} = \mu \alpha_i \beta_j \tag{3}$$

$$b_{ij} = (b_{(i-1,j)} + \alpha_i r_{ij}) + (b_{(i,j-1)} + c_{ij}\beta_j) \tag{4}$$

- Coherent evolution bicluster (E): expression levels are first mapped to labels under certain criteria, such as order or proximity.

The above definitions can be applied to rows, columns or both, but measures are usually used in both dimesions. C biclusters are almost ideal, so algorithms searching for C biclusters usually treats 'constant' as a range of near values by a mapping with coherence evolution.

This bicluster classification presents overlaps. For example, C biclusters on rows and columns (C_{rc}) are included in C biclusters on rows (C_r)and C biclusters on columns (C_c). C biclusters of any type are included in H^+ biclusters and overlap with $H^\times$ biclusters. $H^\pm$ includes them all (Fig. 1). This will be important when comparing biclustering algorithms that search for different kinds of biclusters.

C is the most used group because of direct interpretation in biological data. H^+ biclusters, representing more subtle relations in data are the second group in references. $H^\times$ and $H^\pm$ are rarely used, being their biological relevance difficult to justify or interpret.

2.2 Coherence Measures

Having in mind the different groups of biclusters, we can define measures that determine how constant or how (additive, multiplicative, sign) coherent is our bicluster.

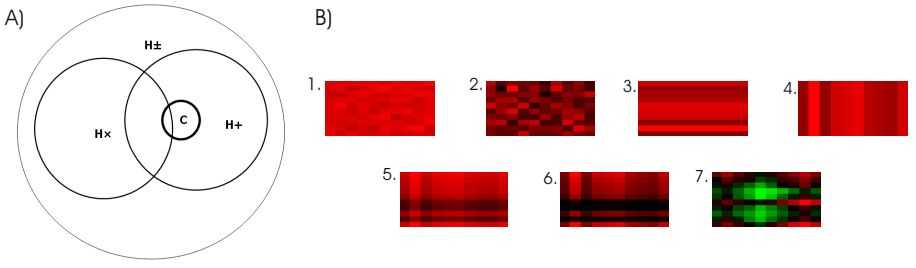


Fig. 1. A) Bicluster sets. Each of the sets is internally divided in row, column and both dimensions biclusters of the corresponding type. B) Heatmaps of different biclusters: 1) C_{rc} bicluster, 2) C_{rc} bicluster with high noise, 3) C_r bicluster, 4) C_c column constant, 5) H^+ bicluster, 6) $H^\times$ bicluster and 7) $H^\pm$ bicluster. 5),6) and 7) become, after row/column transformation, C_r and/or C_c biclusters 3) and 4).

Biclustering algorithms define internally what is considered coherent, but not always under an specific measure or value. Coherence measures can be used to define synthetic biclusters for testing or to check if the results over real data fits the bicluster definition of the algorithm. Constancy by rows of bicluster B ($C_r(B)$) and by columns ($C_c(B)$) are easy to measure by means of Euclidean distance

$$C_r(B) = \frac{1}{n}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sqrt{\sum_{k=1}^{m}(b_{ik}-b_{jk})^2} \tag{5}$$

$$C_c(B) = \frac{1}{m}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}\sqrt{\sum_{k=1}^{n}(b_{ki}-b_{kj})^2} \tag{6}$$

Overall constancy $C_{rc}(B)$ can be derived from $C_r(B)$ and $C_c(B)$:

$$C_{rc}(B) = \frac{nC_r(B)+mC_c(B)}{n+m} \tag{7}$$

The average measure for all the biclusters found by an algorithm is the weighted mean of the measure for each bicluster. These measures, traditionally used to determine cluster compactness will give bad scores for coherent biclusters. To measure coherency, an incremental treatment of the data can be applied to make them 'constant', then applying above formulas to the transformed bicluster $B' = [b'_{ij}]$. In case of H^+:

$$b'_{ij} = b_{ij} - b_{(i-1)j}\ (b_{0j} = 0) \tag{8}$$

$$b'_{ij} = b_{ij} - b_{i(j-1)}\ (b_{i0} = 0) \tag{9}$$

That way, as seen in Fig. 1b, H^+ bicluster becomes C_r and/or C_c bicluster, and can be measured by eqs. 5, 6 and 7. A similar transform can be done with $H^\times$ using division instead of substraction, but now there is necessary to include an exception to avoid divisions by zero:

$$b'_{ij} = b_{ij}/b_{(i-1)j}\ (b_{0j} = 1) \tag{10}$$

$$b'_{ij} = b_{ij}/b_{i(j-1)}\ (b_{i0} = 1) \tag{11}$$

Finally, $H^{\pm}$ has a similar treatment:

$$b'_{ij} = 1 \Leftrightarrow b_{ij} > b_{(i-1)j},\ b'_{ij} = -1\ otherwise.\ (b_{0j} = 1) \tag{12}$$

$$b'_{ij} = 1 \Leftrightarrow b_{ij} > b_{i(j-1)},\ b'_{ij} = -1\ otherwise.\ (b_{i0} = 1) \tag{13}$$

Proximity to zero on all these measures points that the bicluster has the corresponding coherence property. There is no limit in the value they can take, but values above 1.5 usually tells us that coherency is lost (see Section 4 for some practical cases).

3 Validation Indices

Clustering validation indices are divided into three categories [7]: external, internal and relative. External indices measure the similarity between clustering results and a priori knowledge. Internal indices compare the intrinsic structure of data with cluster results. Internal indices are much harder to apply to biclustering than external indices because much of the internal concepts (such as compactness or separation) are not applying to biclusters, where overlapping and coherent variations are usual. Finally relative indices compare different configurations of input parameters and cluster results, trying to find optimal or stable parameters for a given input data.

In the context of biclustering, external validation is mainly used, preferring biological indices to traditional ones. Internal and relative indices are seldom used, because of the non trivial task of adapting biclustering concepts as overlapping and bi-dimensionality to clustering indices.

3.1 Biological External Indices

Biological knowledge used in validations are usually gene annotations as those of Gene Ontology (GO) [2] or KEGG [8]. We will call them external indices because imply information external to the data. Given a bicluster B, we get all (in example) GO terms annotated to any of the genes in B and then apply a statistical significance test to determine if each term appearance is relevant.

Biclustering algorithms presented in [12,3] use GO and/or KEGG enrichment. Other biological knowledge applied in the same way than annotations is related with Transcription Regulatory Networks (TRNs). A TRN is a directed acyclic graph where nodes are genes, and an edge between gene A and gene B means that gene A encodes for a transcription factor protein that transcriptionally regulates (activate or repress) gene B. In this case it is considered the number of genes connected in our bicluster or the average distance between genes in it [12]. It's expected that the number of genes connected will be greater and the average distance lower than in random biclusters, which is checked with a significance test. Another interesting characteristic to check is the number of network motifs (substructures that appear in TRNs [11]) that are included in a bicluster, but it is seldom used in bibliography.

Although useful for the objective of knowledge discovery, biological significance has a major disadvantage as a validation method: biological knowledge is not complete.

When a bicluster does not group known GO/KEGG annotations, or connected genes in a TRN, it may be because it's a bad bicluster, but also because information about TRN connectiveness or GO annotations are not complete. Just as an example, E. coli TRN grew from 424 genes and 577 interactions in 2002 [13] to 1278 genes and 2724 interactions in 2004 [9]. Also statistical significance tests are controversial [6,1].

3.2 Non-biological External Indices

Non-biological external indices are used to check if bicluster results match with previous knowledge of biclusters in the data. They also can be used in comparing biclusters of two different biclustering methods. There are two main techniques to generate external indices: two-matrix and single-matrix techniques.

In case of two-matrix technique, two binary matrices are built, P and R, of size $n \times n$, where n is the number of objects (genes or conditions) of our data. P represents the grouping of objects in the a priori partition and R the grouping in our results. Frow those two matrices, indices are defined, as Rand index, Jaccard coefficient, Minkowski measure or Folkes and Mallows measure [5]. Though the adaptation of two-matrix technique to bi-dimensionality is not very difficult, the concept of overlapping is harder to express with this method, so single matrix is preferred.

Single-matrix technique builds a unique bicluster matrix M of order $p \times r$ where p is the number of biclusters in P and r is the number of biclusters in R. m_{ij} will determine the similarity between the bicluster i of P and the bicluster j of R. A measure of this similarity is F_1 index proposed by Getz et al. [4] and adapted to biclusters by Turner et al. [15]. F_1 is based in the proportion of bicluster i present in bicluster j (sensitivity or module recovery of bicluster i) and the proportion of bicluster j present in bicluster i (specificity or relevance of bicluster i). Note that the sensitivity of bicluster i for j is the specificity of bicluster j for i, and the same with the specificity of i for j, that is the sensitivity of j for i. If g_x is the number of genes in X, c_x the number of conditions in X and $n_x = g_x c_x$; sensitivity, specificity and F_1 are defined as:

$$sensitivity = \frac{(g_{A\cap B})(c_{A\cap B})}{n_B} \tag{14}$$

$$specificity = \frac{(g_{A\cap B})(c_{A\cap B})}{n_A} \tag{15}$$

$$F_1(A,B) = \frac{2(g_{A\cap B})(c_{A\cap B})}{n_A + n_B} \tag{16}$$

When results in R reveal exactly a priori partition P, M will be (if computed with Eq. 16) a square $(p \times p)$, symmetric matrix with $m_{ij} = 1$ if $i = j$ and $m_{ij} = m_{ji} < 1$ otherwise. From M we can get two measures of the overall matching between R and P.

$$S(R,P) = \frac{1}{r}\sum_{i=1}^{r} max_{j=1}^{p}(m_{ij}) \tag{17}$$

$$S(P,R) = \frac{1}{p}\sum_{j=1}^{p} max_{i=1}^{r}(m_{ij}) \tag{18}$$

$S(R,P)$ gives overall bicluster relevance of biclustering R, while $S(R,P)$ gives the module recovery capacity of biclustering R.

3.3 Internal Indices

Internal indices compare intrinsic information about data with the biclustering results. In this case, no a priori information further than the raw data is available. Internal indices are not as precise as external indices, but they are important when a priori information is not available. To avoid the use of internal indices, synthetic data with known structure are built to validate biclustering methods. When applied to real biological data where no a priori information is known, biological tests are used.

An internal index is computed from two matrices just as non-biological external indices. In this case, matrix P contains information about proximity between expression levels of genes or conditions. Now, $P_{ij} = P_{ji} = distance(o_i, o_j)$. Again two pairs of matrices are needed for biclustering, one where o_i are genes and another for conditions. P_{ij} is greater when o_i and o_j are different. R can be built as described for external indices, but inversed so higher values correspond to objects not grouped together. For example $C_{ij} = 1/(1+k)$, where k is the number of times that objects i and j are grouped together. C_{ij} will be in (0,1], being 1 if never grouped together and downing to near 0 if usually grouped. This two matrices can be compared with normalized Hubert statistic:

$$\bar{\Gamma}(C,P) = \frac{\frac{1}{m}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(P_{ij}-\mu_p)(C_{ij}-\mu_c)}{\sigma_p \sigma_c} \tag{19}$$

where n is the number of objects in the matrix, and $m = n(n-1)/2$. μ_p, μ_c are the mean of the matrices and σ_p, σ_p its variances. As with other measures, $\bar{\Gamma}$ index must be computed for the two pair of matrices, then combining as in Eq. 7.

$\bar{\Gamma}$ index and other similar indices, as cophenetic coefficient are less precise than external indices. For example, Jain and Dubes [7] survey different drawbacks of cophenetic coefficient, estimating than even a value of 0.9 will not be enough to assert that there is a good correlation between P and R.

3.4 Relative Indices

Relative indices try to determine the best choice of our algorithm parameters on each particular data set. If we want to compare two algorithms against the same data set, we want to compare its best parametrization for this data set.

However this is a difficult task because of the heterogeneity of the biclustering algorithms and its input parameters. Relative indices use to be external or internal indices, depending on the availability of a priori information from the data. Independently of the index, the procedure is to run the algorithm with different parameter configurations, and compute the index for each one. The parameter configuration with best index is selected as optimal for the data set. Selection of the different parameter configurations is up to the user and is key for the optimal search, so it must represent all the range of possibilities, avoiding deviations.

In clustering, another approach to find the best configuration is to find an stable number of clusters, retrieved by a great number of configurations. From them, we take

the one in the middle of the range, or the one with the best value for a given index. This method is also used in some biclustering validations, usually to find stability when the algorithm has pseudo-random behaviour [3], but not to find optimal initial parameters.

4 Application

4.1 Algorithms

We have applied some of the performance measures discussed to two biclustering algorithms, Bimax [12] and improved Plaid Model of Turner et al [15]. Bimax is one of the most compared biclustering methods, by means of non-biological and biological validation. For example, in [12], non-biological measures are used, but only based in gene dimension because hierarchical clustering was one of the methods compared. Also, in the mentioned comparison only default parameters are used for each algorithm, no parameter optimization is done. Turner plaid model was tested by their authors with different synthetic data sets with three to ten (overlapped in different proportions) biclusters. Turner and Bimax algorithms have never been compared in bibliography.

Both methods have been implemented in R according to the specifications in the corresponding bibliography. Bimax density of 1s against 0s is proved in a range from 1% to 10% (steps of 1%). Turner's t_1 and t_2 parameters are proved as $t_1 = t_2$ in a range from 0.4 to 0.8, with steps of 0.1.

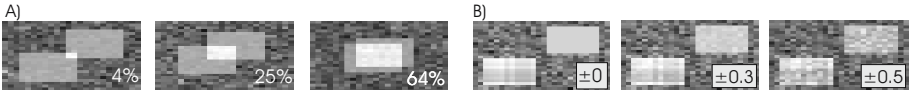


Fig. 2. A) Overlapped constant overexpression biclusters. A low noise has been added to biclusters. Overlapping degree is the same in rows and columns. B) Constant and coherent overexpression biclusters with random noise. Note how noise affects the structure of biclusters, being constancy undistinguishable from coherency with high noise.

4.2 Data Sets

Two sets of synthetic data matrices 100x50 are built. First set of matrices will contain two constant biclusters with overlapping degrees from 0% to 100%, with 10% increments. Second set of matrices have two non-overlapping biclusters, one constant and the other one additive coherent, with normal distribution random noise. Distribution deviation increases from 0 (no noise) to 1, with 0.1 increments. All matrices have a random noise background (see Fig. 2).

4.3 Methods

The proposed test will briefly apply the techniques discussed. First, we will try to find the best parameter choice for each biclustering algorithm in each data set, by means of

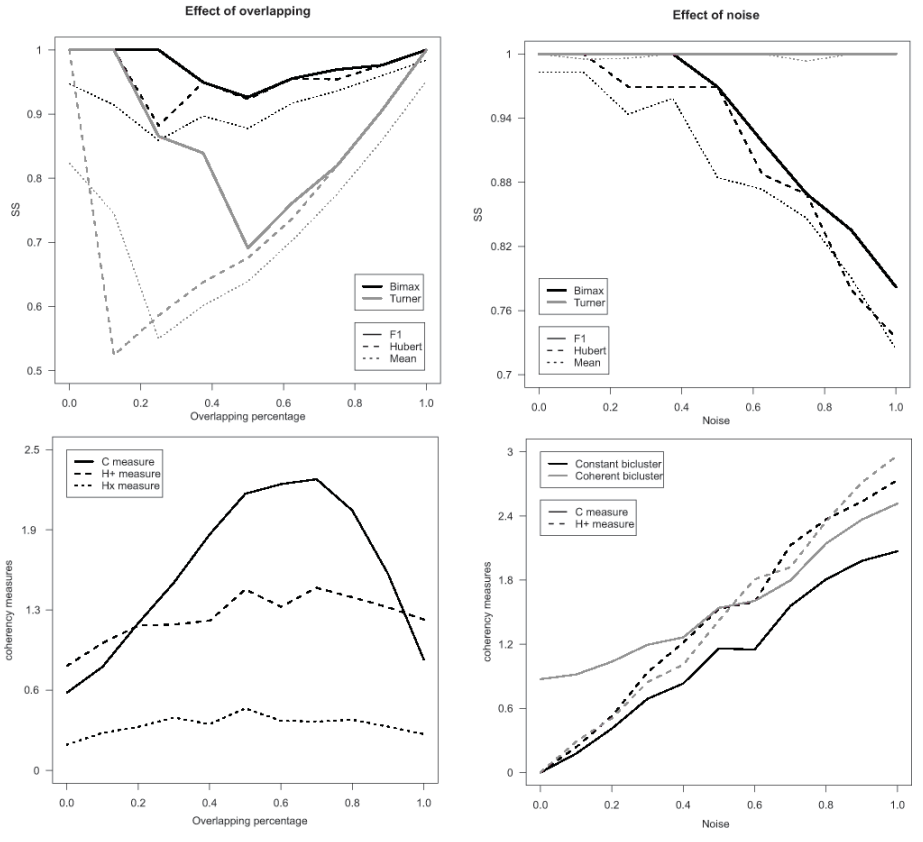


Fig. 3. a) Effect of overlapping in the algorithm and the biclusters. 1) Best *SS* measure achieved by using F_1 and $\bar{\Gamma}$ statistics along with the mean of *SS* for all the proven configurations. 2) Variation in the measures of constancy and coherency with changes in the overlap degree. b1) and b2) As a1) and a2), but representing the effect of the noise in the algorithms and biclusters, respectively.

F_1 measure (comparing against known biclusters) and of $\bar{\Gamma}$ (comparing against proximity matrix). That way, we can compare the performance of $\bar{\Gamma}$ as relative index against an a priori knowledge technique (F_1). Biological significance tests has left out of the scope of this discussion because studies with them are more extended and do not use the measures reviewed here. For known biclusters, constant and coherence measures will be also computed, analyzing its consistency against noise and overlap.

4.4 Results

Fig. 3a-1 presents the mean of sensitivity and specificity (*SS*) of the results of the best configuration given by F_1 and $\bar{\Gamma}$ (or Hubert statistic). F_1 will give the best configuration

at all, while $\bar{\Gamma}$ gives the best configuration supposing a priori information is not available. Also, the mean *SS* for all the tested parametrizations is given. With the appropriate parameter choice, Bimax finds a high percentage of row and columns present in biclusters embedded, even (sometimes) finding the exact biclusters without finding spurious biclusters ($SS = 1$). Performance is lower when overlapping is around 50%, being higher when biclusters are nearly separated or are almost the same. *SS* value of parameter configuration chosen by $\bar{\Gamma}$ measure is obviously worse, but still have better configurations than average. Turner algorithm has lower performance than Bimax. The pruning phase included to improve plaid model fails when trying to prune overlapped parts of the biclusters.

Overlapping effect on biclusters measures is represented in Fig. 3a-2. Because of additive overlapping, intersecting expression levels are higher than non-intersecting, so constant structure is lost with overlapping, in favor of coherent structure.

In Fig. 3b-1 we can see how Bimax performance is sensible to noise when it exceeds 0.4 deviations. Bimax discretization threshold is the responsible of this downgrading. On the other hand, Turner algorithm is not affected by noise, recovering data even in the most noisy cases. Again, $\bar{\Gamma}$ statistic does not give the best configuration in each case, but is better than average. About constancy and coherence measures (Fig. 3 b-2), the measures increase with noise, revealing how structure is eventually lost. Additive coherent bicluster has lower (better) H^+ measure than *C* measure, as expected. Note how H^+ measures increase with noise until, eventually, surpassing *C* measure and coinciding with Bimax performance downgrade.

5 Conclusions and Future Work

Due to the variation and drawbacks of validation indices, the best way to analyze biclustering performance is to use them exhaustively, generating a framework that will define bicluster specific measures (relative, internal and external indices), data type definitions (constant, coherent), benchmark algorithms and example (real and synthetic) data sets.

Though external indices use is extended, our approach to relative and internal index application is new. That helps in automatic optimization of biclustering input paremeters, a task seldom considered and critical for obtaining the highest performance. Data type definition exists as discussed, but only constant biclusters have been mathematically measured. We present an approach to measure coherence biclusters by using constant measures and transformation of data matrices.

External and internal indices used as relative indices have been applied to two biclustering algorithms to prove their consistency and capability to generate information about performance and bicluster behavior against noise and overlap, main problems of biclustering on microarrays. The search of the optimal input parameters for biclustering algorithms through $\bar{\Gamma}$ internal index outperforms the static use of recommended values.

Coherence measures have been also proposed and applied, proving helpful in typifying biclusters. Normalization of these measures must be done to help in comparisons between them. We expect to exhaustively prove all these measures (analyzing and comparing existing biclustering algorithms) and present newer ones in future works.

References

1. Anderson, D.R., Burnham, K.P., Thompson, W.L.: Null hypothesis testing: problems, prevalence, and an alternative. Journal of Wildlife Management 64(4), 912–913 (2000)
2. Ashburner, M., Ball, C.A., Blake, J.A., Bolsteing, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Geneontology: tool for the unification of biology the gene ontology consortium. Nature Genetics 25, 25–29 (2000)
3. Carmona-Saez, P., Pascual-Marqui, R.D., Tirado, F., Carazo, J.M., Pascual-Montano, A.: Biclustering of gene expression data by non-smooth non-negative matrix factorization. BMC Bioinformatics 7(78) (2006)
4. Getz, G., Levine, E., Domany, E.: Coupled two-way clustering analysis of gene microarray data. Proc. Natural Academy of Sciences 97(22), 12079–12084 (2000)
5. Halkidi, M., Batisfakis, Y., Vazirgiannis, M.: On clustering validation techniques. Journal of Intelligent Information Systems 17(2/3), 107–145 (2001)
6. Hubbard, R.: Why we don't really know what "statistical significance" means: a mayor educational failure. Journal of Marketing Education 28, 114–120 (2006)
7. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs (1988)
8. Kanehisa, M., Goto, S.: Kegg: Kyoto encyclopedia of genes and genomes. Nucleic Acids Research 28(1), 27–30 (2000)
9. Ma, H.-W., Kumar, B., Ditges, U., Gunzer, F., Buer, J., Zeng, A.-P.: An extended transcriptional regulatory network of escherichia coli and analysis of its hierarchical structure and network motifs. Nucleic Acids Research 32(22), 6643–6649 (2004)
10. Madeira, S., Oliveira, A.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Transactions of Computational Biology and Bioinformatics 1(1), 24–45 (2004)
11. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science 298, 824–827 (2002)
12. Prelic, A., Bleuer, S., Zimmermann, P., Wille, A., Bhlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9), 1122–1129 (2006)
13. Shen-Orr, S.S., Milo, R., Mangan, S., Alon, U.: Network motifs in the transcriptional regulation network of escherichia coli. Nature Genetics 31, 64–68 (2002)
14. Turner, H., Bailey, T., Krzanowski, W.: Improved biclustering of microarray data demonstrated through systematic performance tests. Computational Statistics and Data Analysis 48, 235–254 (2003)
15. Turner, H.L., Bailey, T.C., Krzanowski, W.J., Hemingway, C.A.: Biclustering models for structured microarray data. IEEE/ACM Transactions on Computational Biology and Bioinformatics 2(4), 316–329 (2005)

Capturing Heuristics and Intelligent Methods for Improving Micro-array Data Classification

Andrea Bosin, Nicoletta Dessì, and Barbara Pes

Università degli Studi di Cagliari, Dipartimento di Matematica e Informatica,
Via Ospedale 72, 09124 Cagliari, Italy
andrea.bosin@dsf.unica.it,{dessi,pes}@unica.it

Abstract. Classification of micro-array data has been studied extensively but only a small amount of research work has been done on classification of micro-array data involving more than two classes. This paper proposes a learning strategy that deals with building a multi-target classifier and takes advantage from well known data mining techniques. To address the intrinsic difficulty of selecting features in order to promote the classification accuracy, the paper considers the use of a set of binary classifiers each of ones is devoted to predict a single class of the multi-classification problem. These classifiers are similar to local experts whose knowledge (about the features that are most correlated to each class value) is taken into account by the learning strategy for selecting an optimal set of features. Results of the experiments performed on a publicly available dataset demonstrate the feasibility of the proposed approach.

1 Introduction

A challenging approach to predict the outcome of certain biological events involves using micro-array experimental data for classifying tumors as either benign or malignant. The general goal of this research line is to turn a qualitative diagnosis by a pathologist into a quantitative diagnosis while leading to a finer understanding of cancer mechanisms.

Currently, each and every micro-array experiment requires measuring several thousand gene fragments and inducing classifiers over them. The classification difficulty depends greatly on the data provided by DNA micro-arrays as they are of high dimensionality while the collected samples are small in number. To simplify, a micro-array dataset is a matrix of m-dimensional observations (i.e. one for each patient) where each dimension is a feature i.e. a measurable property of a specific gene that is designed as gene-expression profile. The class is a label associated to the observation meaning that all samples can be classified as belonging to one of different classes.

While significant progress has been made in the development of machine learning methods for binary classification [1] [2] [3], finding good classifiers is known to be a difficult task when gene expression profiles are used as complex biomarkers defining many different classes of cancer (multi-target classification) and only a small amount of research work has been done on classification involving more than two classes.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 790–799, 2007.

Specifically, difficulties arise when the micro-array dataset exhibits one class that is much more clearly characterized than others (such as [4] or [5] which analyze different tumor types). This makes it impossible to determine which genes are responsible of each single pathology (class value) because the feature selection results in a set that might be most representative of only one (or some, not all) class value.

The purpose of this paper is to present a method for micro-array multi-target classification that tries to overcome the above problems by taking into account two major concerns: the learning strategy and the acquisition of knowledge from specialized binary classifiers. The paper separates these concerns by proposing a learning strategy that deals with building a multi-target classifier and takes advantage from well known data mining techniques (Naïve Bayes [6], Support Vector Machines [7], k-Nearest Neighbor [8]). Focused on the feature selection process, this strategy provides an iterative method that determines at each step which features to acquire next in order to find an optimal set of features. A major contribute of this paper is in the second concern that deals with proposing the use of a set of binary classifiers each of ones is devoted to predict a single class of the multi-classification problem. From this point of view, the binary classifiers are similar to local experts whose knowledge is taken into account by the learning process that collects and takes benefit of information concerning the optimal set of features that each single binary classifier selects.

The paper is organized as follows. Section 2 briefly summarizes some related works. Section 3 illustrates our learning strategy, whose validation is asserted by experiments presented in Section 4. Finally, Section 5 presents a brief discussion as well as concluding remarks.

2 Related Work

Building a predictive model (i. e. a classifier that is expected to discriminate normal from cancer tissues or to distinguish among different classes of tumors) presents multiple challenges, because the micro-array data include a large number of gene expression values per experiment (several thousands of features), and a relatively small number of samples (a few dozen of patients). Most features being irrelevant to the problem at hand, the micro-array data exhibit a high degree of noise: a large number of features in input to the learning algorithms may turn them to build inefficient classifiers with the additional charge of memory and time consumption. Moreover, correlation between feature sets results in the counter effect of over-fitting [3]: that is creating classifiers that may not generalize well to new data from the same type and distribution, despite their excellent accuracy on the training set.

A number of studies have shown that accurate classification of micro-array data can be made using a reduced number of genes [3][4][9] indicating that it is more important to explore data and utilize independent features to train classifiers, rather than increase the number of features we use. Additionally, the identification of discriminatory genes is of fundamental and practical interest since medical diagnostic tests may benefit from the examination of a small subset of relevant genes.

The underlying distribution of the features being not known, the feature selection process originates ambiguity in deciding which group of features constitutes an optimal set. The central question of the “minimum informative subset problem” still

remains an active focus of micro-array research and is a challenging task because the number of possible feature subsets increases exponentially with the number of features, making exhaustive search impractical.

Along with this central question there is a range of other questions, compactly summarized in [10], and it is not clear how to proceed with feature selection, knowing that under different conditions the selected feature subsets will likely be different depending on sample sizes for the validation process and on different selection methods. Furthermore, the small sample size and high dimensionality of the data constrain the possibility of properly validating the chosen classification model and different classifiers perform differently on micro-arrays, with dataset sparsity as the major contributor for the differences [11]. Selecting simple classifiers that need minimal parameter tuning seems to be the appropriate approach, independently of data complexity and especially for small sample sizes.

3 The Learning Strategy

A multi-target classification problem can be treated directly or decomposed into several binary classification problems. However, it has been observed [12] that the direct application of a learning strategy to the multi-target problem may result in an over-representation of the abundant and/or easily separable classes. To circumvent the above mentioned limitation, we consider breaking the original M-class problem into a set of binary sub-problems (one for each class) and performing classification by training and combining these binary classifiers with respect to some criterion.

Because we are interested in separating each pathology from all the others, the proposed strategy adopts the one-versus-rest (1-vs-r) classification schema that provides M binary classifiers, each of ones is trained for distinguishing between a given class and the M-1 other remaining classes whose instances are considered as negative examples. According to this schema, the first step of the learning strategy decomposes the original dataset into M sub-datasets, each separating the instances of a given class from the rest of the classes. Then, a filter is applied to each sub-dataset that orders the features by their rank within each class, a high score being indicative of a relevant variable. Specifically, the ranking procedure concentrates on the correlation of each gene with only one class value at a time, against all the others, and originates M sets of ranked genes, each of ones is correlated only with one given pathology (this is much more interesting also from a medical point of view): irrelevant features are excluded from the classification task, thereby reducing both the noise of the dataset as well as the time needed to perform the classification.

This pre-processing step, which is independent from the classification task, is based on some statistical criteria (i.e. MDL, χ^2) and provides a basis for numerically weighting the variables individually, but the next crucial step is the selection of a smaller number of highly specific features, i.e. an “optimal” set of features to employ in learning each binary classifier. The classification process dealing with a large number of variables, it is computationally intractable to search the whole space of feature subsets and one has to settle for approximations of the minimal optimal set of features that significantly improves the learning algorithm’s performance. Consequently, a

feature selection process defines both a strategy to search the space of possible feature subsets and a measure for assessing the goodness of the selected subset.

Our approach considers each binary class decision as problem instance and selects features for it separately. It results in M-separated processes of feature selection each of ones first considers the N top-ranked attributes for the sub-dataset under examination. The basic idea is to select variables step by step according to their predictive power and using a sequential forward selection that starts the search with an empty variable subset. During each single step, one or more variables are considered for the inclusion in the subset using as criterion the performance of a classifier built with a new subset that is obtained by the inclusion of the considered variables**.**

Table 1 depicts a general schema of the learning strategy that is applied to each single sub-dataset.

Table 1. A general schema of the learning strategy

1	Consider a single sub-dataset R and rank all its features according to a statistical criterion
2	Select the N top-ranked features (e.g. start with N = 1)
3	Build a binary classifier with the N top-ranked features
4	Test classifier accuracy (on an independent test dataset)
5	Extend the feature subset by adding the next k top-ranked features (e.g. k = 1) and put N = N + k
6	Repeat steps 2 to 5 and stop if the accuracy has not increased according to some stopping criterion or after a fixed number of iterations

Because just a small number of variables is needed for separating the micro-array data, a ranking criterion based on classification success rate can distinguish between the top-ranked variables [3]. However, the filter methods and the classification algorithms are not directly relevant for this study: the question we address here is that of comparing the above mentioned nested classifiers and proposing a stopping criterion for halting iterations.

Measured in terms of false positive (FP) classification rate and false negative (FN) classification rate, the classification accuracy is usually still the only measure used for evaluating the performance of micro-array classifiers that are based on data mining techniques. As well, measures such recall and precision are popular metrics employed in data mining applications. Moreover, it is quite common to monitor the tradeoff between true positives (TP) and false positives (FP) by graphical means such as the Receiver Operating Characteristic (ROC) curve that shows FP on the x-axis while TP is plotted along the y-axis. The visualization of the classifier performance is one of the attractive features of ROC analysis that is useful for comparing the relative performance among different classifiers while the area under the ROC curve (AUC) provides another approach for evaluating which model is better on average.

A recent work [13] outlines that ROC curves are inadequate for the needs of data mining research in several significant respects and demonstrate the validity of the cost

curves [14] as a graphical mean for overcoming these deficiencies. Based on the point/line duality between ROC space and cost space, this graphical technique (best detailed in [15]) plots, along the x-axis, the *probability cost* $p_C[+]$, defined as

$$p_C[+] = p[+]C[+|-] / (p[+]C[+|-] + p[-]C[-|+]) \quad (1)$$

and, along the y-axis, the *normalized expected cost* NEC, defined as

$$NEC = FN \cdot p_C[+] + FP \cdot (1 - p_C[+]) \quad (2)$$

By p[+] we denote the probability of the positive class, and by C[+|-] the cost of predicting + when the instance is actually - (and by C[-|+] the reverse). Basically, the probability cost $p_C[+]$ is a distorted version of p[+] based on the cost notion, while the normalized expected cost measures the classifier performance weighting each classification error by the corresponding cost. Both the probability cost and the normalized expected cost range from 0 to 1.

We experimented cost curves as a measure for comparing the performance of a family of nested classifiers and for selecting the classifier that has the lowest cost. Specifically, we modified the proposed strategy by evaluating, at the end of each step, the cost curves of the actual and the previous classifier: the vertical distance between the cost curves directly indicates the performance difference between them. The iteration is carried on as long as the cost difference is greater than a fixed threshold and the variable subset corresponding to the lowest cost classifier is assumed as optimal subset.

This approach assumes that each binary classifier is best inside certain subsets of the whole feature domain. However, the problem is to integrate this information coming from multiple independent binary classifiers that are similar to specialized local experts. To this end, a new approach is here explored that considers to treat directly the multi-class problem by a multi-classifier learnt on a "globally optimal" subset of features that results by joining all the optimal subsets generated by the presented learning strategy.

This "knowledge-based" multi-classifier is a mechanism to combine information received from several sources while the learning strategy is employed just for feature selection i.e. and for achieving knowledge about genes featuring each single class. Because only the genes really relevant to each pathology are involved in multi-classification, it is expected that the accuracy of a "knowledge-based" multi-classifier may be increased by the knowledge of local binary classifiers that are most reliable for specific domains.

4 Experiments

The Acute Lymphoblastic Leukaemia dataset [16] (ALL, from now) has been used as a test-bed for the experiments presented in this section. ALL consists of 327 samples (specifically, 215 training and 112 test samples), each one described by the expression level of 12558 genes. 7 classes are involved in total, i.e. all known ALL sub-types (T-ALL, E2A-PBX1, TEL-AML1, BCR-ABL, MLL, Hyperdip > 50) and a generic class OTHERS, that groups all samples not belonging to any of the previous sub-types.

For all the experiments we carried on, we applied the χ^2 statistics, as a ranking criterion, and the following popular classification algorithms: Naïve Bayes (NB) [6], Support Vector Machines (SVM) [7], k-Nearest Neighbor (k-NN) [8]. All the experiments have been carried out using the Weka machine learning software package [15] and a software tool based on cost curves [13].

Fig. 1 shows the accuracy of SVM, NB and k-NN multi-classifiers as a function of the number of features when the multi-target classification problem is treated directly. This means that feature selection is performed by measuring the correlation of each gene simultaneously with all class values. As can we see, the behavior of all classifiers is similar: the accuracy has some initial oscillation and a large number of features (between 750 and 1500) is necessary to reach a plateau and achieve the maximum accuracy, in agreement with recent literature [17] [18]. Additionally, we cannot discriminate which features are mostly correlated to a specific class value.

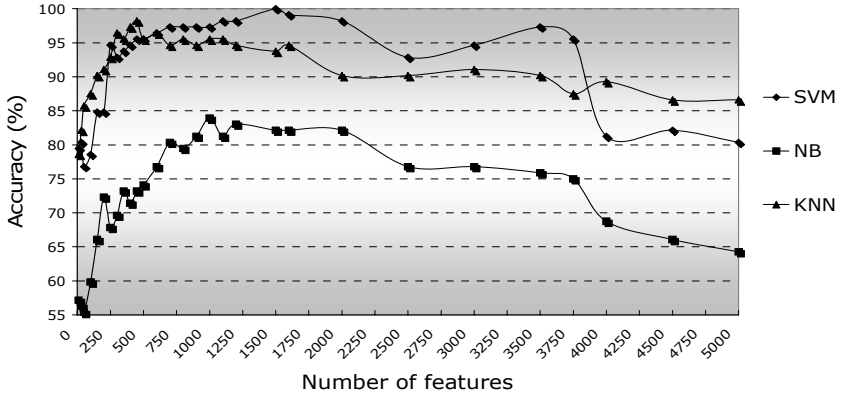


Fig. 1. Accuracy of SVM, NB and k-NN multi-classifiers

In the following, results are presented using the proposed learning strategy. As a first step, the original training set was decomposed into 6 binary sub-datasets by separating the instances of each single sub-type from the rest whose instances were considered negative examples. After data preparation, we applied the ranking procedure that discriminates, for each sub-dataset, the set of features most strongly related to each target class. This resulted in 6 sets of ranked features ordered by their rank within each class. Then, for each classification algorithm (i.e. SVM, NB and k-NN), we applied the learning strategy described in Section 3.

Fig. 2 shows, in the left panel, the cost curves for two nested binary classifiers learnt with 3 (dashed line) and 9 features (solid line) when the SVM algorithm is applied to the sub-dataset that separates the class BCR-ABL. The cost difference is clearly observable and shows that the accuracy increases while augmenting the number of features. On the contrary, the right panel in Fig. 2 shows an accuracy reduction when a new feature is added in building the family of nested classifiers for the sub-dataset that separates the class T-ALL. Specifically, the cost curves depict the performance of the SVM classifier learnt with the first top-ranked feature (dashed line) and the first two top-ranked features (solid line).

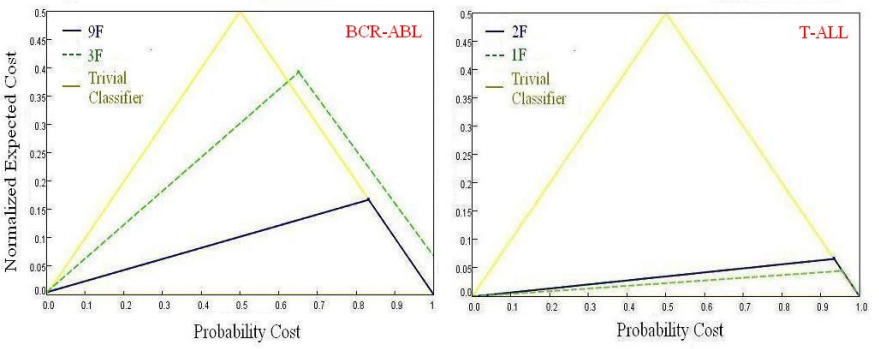


Fig. 2. Cost curves for two SVM nested classifiers learnt on the sub-dataset that separates the class BCR-ABL (left panel) and the class T-ALL (right panel)

Table 2 shows the resulting optimal number of features respective of each single class as well as the accuracy of the binary classifiers built on these optimal subsets (in brackets).

Table 2. Optimal number of features for each ALL sub-type and best accuracy for each binary classifier (in brackets)

	T-ALL	***E2A-PBX1***	***TEL-AML1***	***BCR-ABL***	***MLL***	***Hyperdip>50***
SVM	1 *(100 %)*	1 *(100 %)*	12 *(100 %)*	9 *(99,1 %)*	3 *(100 %)*	5 *(96,4 %)*
NB	1 *(100 %)*	1 *(100 %)*	14 *(100 %)*	10 *(97,3 %)*	16 *(99,1 %)*	12 *(96,4 %)*
k-NN	1 *(100 %)*	1 *(100 %)*	12 *(100 %)*	3 *(97,3%)*	3 *(100 %)*	9 *(95,5 %)*

Results in Table 2 show that binary classifiers achieve their maximum accuracy with very few features, in that outperforming standard multi-classifiers (Fig. 1). In particular, only one gene is sufficient to perfectly discriminate T-ALL and E2A-PBX1 sub-types, irrespective of the adopted classification algorithm. No error occurs in TEL-AML1 classification too, even if a higher number of features (12-14) is required. Most misclassifications occur in discriminating BCR-ABL and Hyperdyp>50, suggesting a less sharp genetic characterization of these sub-types.

Finally, Table 3 summarizes results when the proposed "knowledge-based" approach is applied, i.e. when the multi-class problem is treated directly using a "globally optimal" subset of features that results by joining all the optimal subsets previous generated.

To approximately evaluate the statistical significance of the above accuracy measure, we constructed a binomial confidence interval as suggested in [19]; specifically, last column of Table 3 reports, in brackets, the intervals relative to a 95% confidence level. As shown by these results, the proposed "knowledge-based" approach turns out to be much more accurate than standard multi-classifiers depicted in Fig.1. Indeed, the accuracy of a standard SVM multi-classifier is only 80,4% (72,1 ÷ 86,7%) with 30

Table 3. Optimal number of features and classification accuracy for SVM, NB and k-NN knowledge-based multi-classifiers

	Number of features	*Accuracy*
SVM	31	96,4 % (91,1 ÷ 98,6%)
NB	54	86,6 % (79,1 ÷ 91,7%)
k-NN	29	93,8 % (87,7 ÷ 97,0%)

features, while a knowledge-based multi-classifier achieves 96,4% (91,1 ÷ 98,6%) with 31 features, as witnessed by confusion matrices in Tables 4. The statistical significance of the observed difference has been further proved by the application of the McNemar'test [20], which is recommended [19][21] for the cases where the learning algorithm is run only once (i.e. without any form of resampling).

Table 4. Confusion matrices of standard (**A**) and knowledge-based (**B**) SVM multi-classifiers

A. *Standard multi-classification*

actual class \ predicted class	**a**	**b**	**c**	**d**	**e**	**f**	**g**
a	1	0	0	0	3	0	2
b	0	9	0	0	0	0	0
c	0	0	12	0	10	0	0
d	0	0	1	4	1	0	0
e	2	0	1	0	22	0	2
f	0	0	0	0	0	15	0
g	0	0	0	0	0	0	27

30 features, accuracy ***= 80,4%***

B. *"Knowledge-based" multi-classification*

actual class \ predicted class	**a**	**b**	**c**	**d**	**e**	**f**	**g**
a	5	0	1	0	0	0	0
b	0	9	0	0	0	0	0
c	0	0	21	0	1	0	0
d	0	0	0	6	0	0	0
e	0	0	1	0	26	0	0
f	0	0	0	0	0	15	0
g	0	0	0	0	1	0	26

31 features, accuracy = ***96,4%***

a = BCR-ABL, **b** = E2A-PBX1, **c** = Hyperdip>50,
d = MLL, **e** = OTHERS, **f** = T-ALL, **g** = TEL-AML1

Analogously, the performance of NB and k-NN multi-classifiers (whose confusion matrices are omitted for sake of space) greatly improves when the knowledge-based approach is adopted. Specifically, NB achieves 55,4% (46,2 ÷ 64,3%) with 50 features selected in the standard way, while its accuracy is 86,6% (79,1 ÷ 91,7%) when an optimal set of 54 features is selected according to the proposed knowledge-based heuristic. Similarly, the standard k-NN multi-classifier achieves 82,1% (74,0 ÷ 88,1%) with 30 features, while the accuracy of the knowledge-based k-NN multi-classifier is 93,8% (87,7 ÷ 97,0%) with 29 features.

5 Discussion and Concluding Remarks

The presented learning strategy has been validated by different experiments. The first experiment (Fig.1), here referred to as standard multi-classification, highlights the intrinsic weakness of a feature selection process where the correlation of each gene

with all class values is simultaneously measured. Related difficulties are also reported by recent literature [17] [18] [22].

The second experiment (Fig. 2 and Table 2) reveals that a 1-vs-r decomposition is much more effective than a single multi-classifier. Our results can be compared with [4] where six different heuristics for feature selection are explored by learning NB, SVM and k-NN binary classifiers on the ALL dataset. In [4] the best feature selection heuristic ranks attributes according to their entropy and selects features "having an entropy value less than 0.1 if these exist, or the 20 features with the lowest entropy values otherwise", up to a maximum of 20 features for each binary classifier. The resulting NB, SVM and k-NN models are then combined according to a hierarchical scheme and respectively misclassify 7, 5 and 4 samples. It is important to observe that in [4] the threshold of 20 used to cut off top-ranked features is an arbitrary number, whereas our learning strategy enables to find, for a given ranking method and a given classification algorithm, the "optimal" cut off point.

The paper goes further in this improving classification by proposing a new approach referred to as "knowledge-based" multi-classification. As shown by the third experiment (Table 3 and Table 4), this approach turns out to be much more accurate than standard multi-classifiers, with the same number of features. [5] is the only work where a similar heuristic is adopted, but on a different case study. However, in [5] the global subset of features is built by taking an equal number of genes for each class, hence requiring more genes than in our approach. Indeed, as witnessed by our analysis, the optimal number of features can be different for different class values, suggesting a different strength in genetic characterization of cancer sub-types.

Our approach is tailored to capture genetic sub-type specificity, enabling to sensibly reduce the total number of features involved in multi-classification and returning only the features that are really relevant. As future extension, we plan to validate our strategy on different multi-cancer datasets, to obtain more insights on genetic mechanisms underlying cancer sub-type differentiation. In particular, the effectiveness of the proposed knowledge-based approach will be further investigated in the context of poorly differentiated cancer-sub-types, where the identification on the genes responsible of each sub-type is of crucial importance.

Acknowledgments

The authors are very grateful to Robert Holte for the useful suggestions and for the freely distribution of the cost curve software tool that has been used for the cost based analysis.

References

1. Golub, T.R., et al.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
2. Alizadeh, A.A., et al.: Distinct Types of Diffuse Large B-cell Lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
3. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene Selection for Cancer Classification using Support Vector Machines. Machine Learning 46(1-3), 389–422 (2002)

4. Liu, H., et al.: A Comparative Study on Feature Selection and Classification Methods Using Gene Expression Profiles and Proteomic Patterns. Genome informatics 13, 51–60 (2002)
5. Piatetsky-Shapiro, G., et al.: Capturing Best Practice for Microarray Gene Expression Data Analysis. In: SIGKDD 2003, Washington, USA (August 24-27, 2003)
6. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian Network Classifiers. Machine Learning 29, 131–161 (1997)
7. Vapnik, V.: Statistical Learning Theory. Wiley-Interscience, New York (1998)
8. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13, 21–27 (1967)
9. Tao, L., et al.: A Comparative Study on Feature Selection and Multiclass Classification Methods for tissue classification based on gene expression. Bioinformatics 20(15) (2004)
10. Simon, H.: Supervised analysis when the number of candidate features greatly exceeds the number of cases. SIGKDD Explorations 5(2), 31–36 (2003)
11. Somorjai, R., et al.: Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, cavets, cautions. Bioinformatics 19(12) (2003)
12. Forman, G.: An extensive empirical study of feature selection metrics for text classification. JMLR 3, 1289–1306 (2003)
13. Drummond, C., Holte, R.C.: Cost Curves: An improved Method for Visualizing Classifier Performance. Machine Learning Journal 65(1) (October 2006)
14. Drummond, C., Holte, R.C.: Explicitly Representing Expected Cost: An Alternative to ROC Representation. In: Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining, pp. 198–207 (2000)
15. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Elsevier, Amsterdam (2005)
16. St. Jude Children's Research Hospital, http://www.stjuderesearch.org/data/ALL1/
17. Mukherjee, S.: Classifying Microarray Data Using Support Vector Machines, Understanding And Using Microarray Analysis Techniques: A Practical Guide. Kluwer Academic Publishers, Boston, MA (2003)
18. Statnikov, A., et al.: A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. Bioinformatics 21(5) (2005)
19. Dietterich, T.G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation 10, 1895–1924 (1998)
20. Everitt, B.S.: The analysis of contingency tables. Chapman and Hall, London (1977)
21. Demsar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. Journal of Machine Learning Research 7, 1–30 (2006)
22. Bosin, A., Dessì, N., Pes, B.: A Cost-Sensitive Approach to Feature Selection in Micro-Array Data Classification. In: Proceedings of WILF 2007, Portofino, Italy (July 2007)

Classification of Microarrays with kNN: Comparison of Dimensionality Reduction Methods

Sampath Deegalla[1] and Henrik Boström[2]

[1] Dept. of Computer and Systems Sciences,
Stockholm University and Royal Institute of Technology,
Forum 100, SE-164 40 Kista, Sweden
si-sap@dsv.su.se
[2] School of Humanities and Informatics,
University of Skövde,
P.O. Box 408, SE-541 28, Skövde, Sweden
henrik.bostrom@his.se

Abstract. Dimensionality reduction can often improve the performance of the k-nearest neighbor classifier (kNN) for high-dimensional data sets, such as microarrays. The effect of the choice of dimensionality reduction method on the predictive performance of kNN for classifying microarray data is an open issue, and four common dimensionality reduction methods, Principal Component Analysis (PCA), Random Projection (RP), Partial Least Squares (PLS) and Information Gain(IG), are compared on eight microarray data sets. It is observed that all dimensionality reduction methods result in more accurate classifiers than what is obtained from using the raw attributes. Furthermore, it is observed that both PCA and PLS reach their best accuracies with fewer components than the other two methods, and that RP needs far more components than the others to outperform kNN on the non-reduced dataset. None of the dimensionality reduction methods can be concluded to generally outperform the others, although PLS is shown to be superior on all four binary classification tasks, but the main conclusion from the study is that the choice of dimensionality reduction method can be of major importance when classifying microarrays using kNN.

1 Introduction

Microarray gene-expression technology has spread across the research community with immense speed during the last decade [1]. Being able to effectively learn from data generated through this technology is important for many reasons, including allowing for early accurate diagnoses which might lead to proper choice of treatments and therapies [2,3]. On the other hand, this type of high-dimensional data, often involving thousands of attributes, creates challenges for many learning algorithms, including the well-known k-nearest neighbor classifier (kNN) [4].

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 800–809, 2007.

The kNN has a very simple strategy as a learner: instead of generating an explicit model, it keeps all training instances. A classification is made by measuring the distances from the test instance to all training instances, most commonly using the Euclidean distance. Finally, the majority class among the k nearest instances is assigned to the test instance. This simple form of kNN can however be both inefficient and ineffective for high-dimensional data sets due to presence of irrelevant and redundant attributes. Therefore the classification accuracy of kNN often decreases with an increase in dimensionality. One possible remedy to this problem that earlier has shown to be successful is to use dimensionality reduction [5].

The kNN has earlier been demonstrated to allow for successful classification of microarrays [2] and it has also been shown that dimensionality reduction can further improve the performance of kNN for this task [5]. However, it is an open question if the choice of dimensionality reduction technique has any impact in the performance, and for this purpose, four commonly employed dimensionality reduction methods are compared in this study when used in conjunction with kNN for microarray classification.

The organization of the paper is as follows. In the next section, we briefly present the four dimensionality reduction methods used in the study. In section 3, details of the experimental setup are provided, and the results of the comparison on eight microarray data sets are given. Finally, we give some concluding remarks and outline directions for future work.

2 Dimensionality Reduction

2.1 Principal Component Analysis (PCA)

PCA uses a linear transformation to obtain a simplified data set retaining the characteristics of the original data set.

Assume that the original matrix contains d dimensions and n observations and that one wants to reduce the matrix into a k dimensional subspace. This transformation can be given by [6]:

$$Y = E^T X \tag{1}$$

where $E_{d\times k}$ is the projection matrix containing k eigen vectors corresponding to the k highest eigen values, and $X_{d\times n}$ is the mean centered data matrix.

2.2 Random Projection (RP)

By RP, the original data set is transformed into a lower dimensional subspace by using a random matrix [7,8].

Assume that one wants to reduce the d dimensional data set into a k dimensional set where the number of instances are n. The transformation is then given by:

$$Y = R X \tag{2}$$

where $R_{k\times d}$ is the random matrix and $X_{d\times n}$ is the original data matrix. The original idea behind the RP is based on the Johnson-Lindenstrauss lemma (JL)

[9] which states that n points can be projected from $R^d \rightarrow R^k$ while preserving the Euclidean distance between the points within an arbitrarily small factor. For more details on the method, see [8].

This random matrix can be created in several ways and the one we have used is introduced by Achlioptas [10], by which the random matrix is generated as follows.

$$r_{ij} = \begin{cases} +\sqrt{3} \text{ with } P_r = \frac{1}{6}; \\ 0 \text{ with } P_r = \frac{2}{3}; \\ -\sqrt{3} \text{ with } P_r = \frac{1}{6}. \end{cases} \quad (3)$$

2.3 Partial Least Squares (PLS)

PLS was originally developed within the social sciences and has later been used extensively in chemometrics as a regression method [11]. It seeks for a linear combination of attributes whose correlation with the class attribute is maximized. In PLS regression the task is to build a linear model, $\bar{Y} = BX + E$, where B is the matrix of regression coefficients and E is the matrix of error coefficients. In PLS, this is done via the factor score matrix $Y = WX$ with an appropriate weight matrix W. Then it considers the linear model, $\bar{Y} = QY + E$, where Q is the matrix of regression coefficients for Y. Computation of Q will yield $\bar{Y} = BX + E$, where $B = WQ$. However, we are interested in dimensionality reduction using PLS and used the SIMPLS algorithm [12,13]. In SIMPLS, the weights are calculated by maximizing the covariance of the score vectors y_a and $\bar{y_a}$ where $a = 1, \ldots, A$ (where A is the selected numbers of PLS components) under some conditions. For more details of the method and its use, see [12,14]

2.4 Information Gain (IG)

Information Gain (IG) can be used to measure the information content in a feature [15], and is commonly used for decision tree induction. Maximizing IG is equivalent to minimizing:

$$\sum_{i=1}^{V} \frac{n_i}{N} \sum_{j=1}^{K} -\frac{n_{ij}}{n_i} \log_2 \frac{n_{ij}}{n_i}$$

where K is the number of classes, V is the number of values of the attribute, N is the total number of examples, n_i is the number of examples having the ith value of the attribute and n_{ij} is the number of examples in the latter group belonging to the jth class.

3 Empirical Study

3.1 Data Sets

The following eight microrarray data sets are used in this study:

– Colon Tumor [16], which consists of 40 tumor and 22 normal colon samples.

- Leukemia [17], which contains 72 samples of two types of leukemia: 25 acute myeloid leukemia (AML) and 47 acute lymphoblastic leukemia (ALL).
- Central Nervous System [18], which consists of 60 patient samples of survivors (39) and failures (21) after treatment of the medulloblastomas tumor (This is data set C from [18]).
- SRBCT [3], which contains four diagnostic categories of small, round blue-cell tumors as neuroblastoma (NB), rhabdomyosarcoma (RMS), non-Hodgkin lymphoma (NHL) and the Ewing family of tumors (EWS).
- Lymphoma [19], which contains 42 samples of diffuse large B-cell lymphoma (DLBCL), 9 follicular lymphoma (FL) and 11 chronic lymphocytic leukemia (CLL).
- Brain [18] contains 42 patient samples of five different brain tumor types: medulloblastomas (10), malignant gliomas (10), AT/RTs (10), PNETs (8) and normal cerebella (4). (This is the data set A from [18].)
- NCI60 [20], which contains eight different tumor types. These are breast, central nervous system, colon, leukemia, melanoma, non-small cel lung carcinoma, ovarian and renal tumors.
- Prostate [2], which consists of 52 prostate tumor and 50 normal specimens.

The first three data sets come from Kent Ridge Bio-medical Data Set Repository [21] and the remaining five from [22]. The data sets are summarized in Table 1.

Table 1. Description of data

Data set	Attributes	Instances	# of Classes
Colon Tumor	2000	62	2
Leukemia	7129	38	2
Central Nervous	7129	60	2
SRBCT	2308	63	4
Lymphoma	4026	62	3
Brain	5597	42	5
NCI60	5244	61	8
Prostate	6033	102	2

3.2 Experimental Setup

We have used Matlab to transform raw attributes to both PLS and PCA components. The PCA transformation is performed using the Matlab's Statistics Toolbox whereas the PLS transformation is performed using the BDK-SOMPLS toolbox [23,24], which uses the SIMPLS algorithm. The WEKA data mining toolkit [15] is used for the RP and IG methods, as well as for the actual nearest neighbor classification.

Both PLS and IG are supervised methods which use class information for their transformations. Therefore, to generate the PLS components for test sets, the

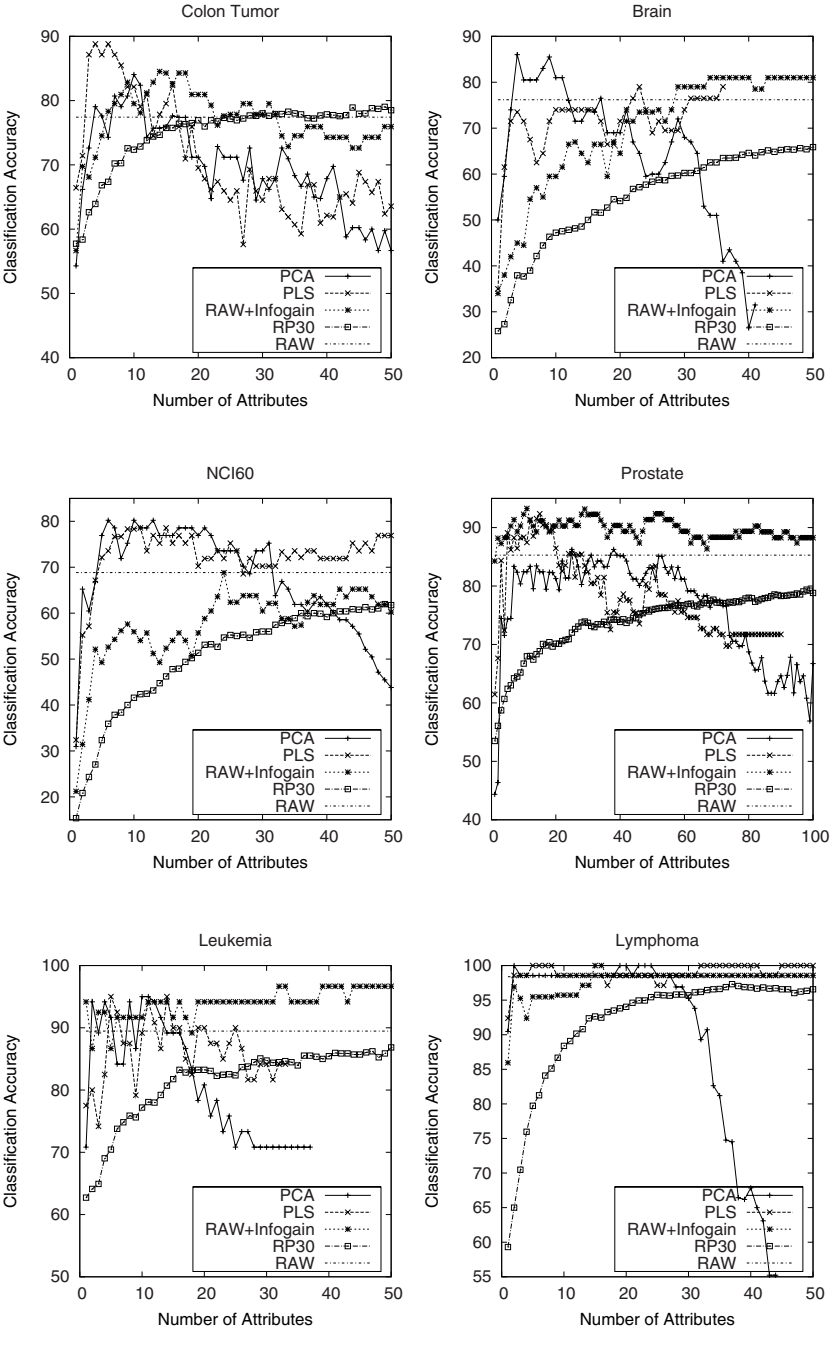


Fig. 1. Predictive performance with the change of numbers of dimensions using PCA, PLS, RP and IG with Nearest Neighbor (IB1) for Colon Tumor, Brain, NCI60, Prostate, Leukemia and Lymphoma data sets

weight matrix generated for the training set has to be used. For IG, attributes in the training set are ranked based on the information content in a decreasing manner and the same attributes are selected for the test set. As earlier explained, attributes generated using RP are of a random nature since a random matrix is used for the transformation. For this reason, we have averaged results of RP from 30 runs to reduce the variance.

The optimal number of neighbors (i.e., k) could be specific to different data sets and dimensionality reduction methods. Therefore, we have investigated the effect of different values of k, namely 1, 3, 5, 7 and 9.

Stratified 10-fold cross validation [15] is employed to obtain measures of accuracy, which has been chosen as the performance measure in this study.

3.3 Experimental Results

The results are summarized in Fig. 1 and Fig. 2. It can be observed that both PLS and PCA obtain their best classification accuracies with relatively few dimensions, while more dimensions are required for IG and many more for RP.

None of the methods turns out as a clear winner, except perhaps PLS on the binary classification tasks. However, all methods outperform not using dimensionality reduction, and the difference in performance between the best and worst method can vary greatly for a particular dataset, leading to the conclusion that the choice of dimensionality reduction to be used in conjunction with kNN for microarray classification can be of major importance.

In most of the cases, simply setting $k = 1$ gives the best result. However, for IG it seems that one should consider choosing higher values for k which improves the classification accuracy by at least 1% for 5 out of 8 datasets. For PCA, the choice of a higher k value yields at least a 1% improvement for 3 out of 8 data sets whereas for PLS, an improvement of at least 1% is obtained for 4 out of 8 datasets.

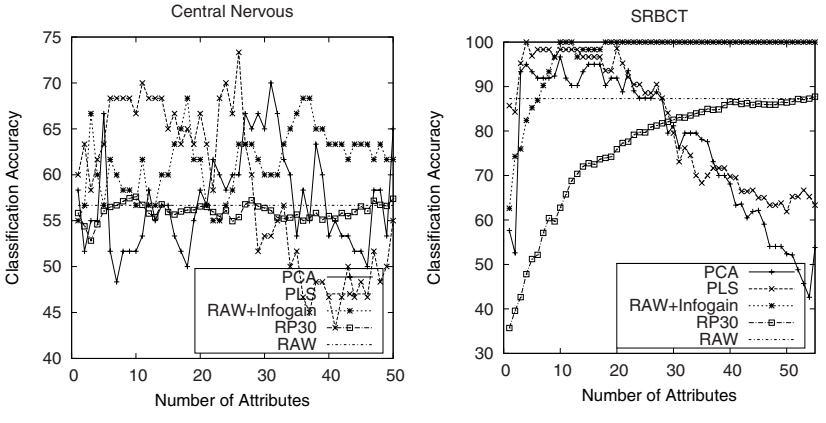


Fig. 2. Predictive performance with the change of numbers of dimensions using PCA, PLS, RP and IG with Nearest Neighbor (IB1) for Central Nervous and SRBCT data sets

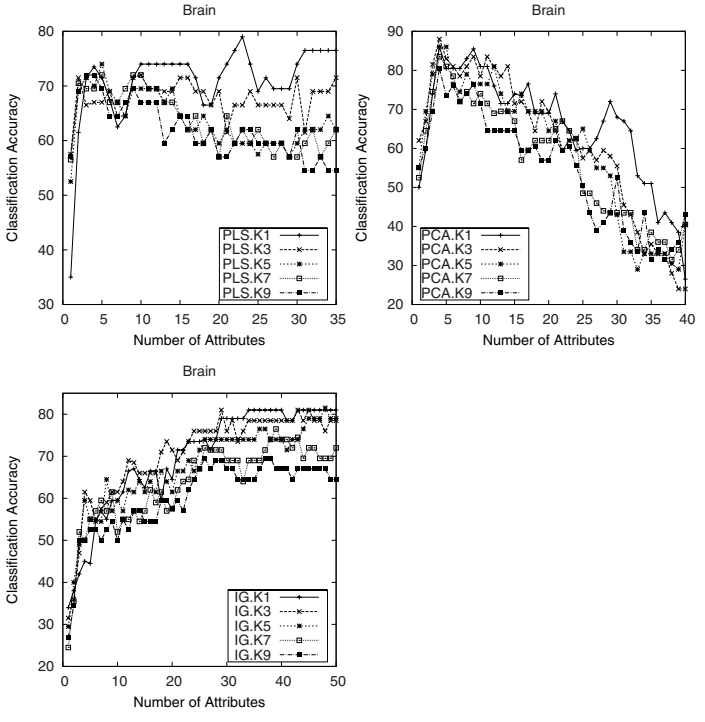


Fig. 3. Predictive accuracy with different k values for nearest neighbor classifier for Brain dataset

Table 2. Order of k values w.r.t averaged accuracy

	Decreasing order of accuracy		
	IG	PCA	PLS
ColonTumor	7,5,9,3,1	5,9,7,3,1	7,9,5,3,1
Leukemia	1,3,5,7,9	1,3,5,7,9	3,1,5,7,9
CentralNervous	7,9,5,1,3	3,7,9,5,1	9,7,5,3,1
SRBCT	3,5,1,7,9	1,9,3,7,5	9,7,5,3,1
Lymphoma	5,9,1,7,5	1,3,5,7,9	1,3,5,7,9
Brain	3,1,5,7,9	1,3,5,7,9	1,3,5,7,9
NCI60	9,7,1,5,3	1,3,5,7,9	1,3,5,7,9
Prostate	3,7,9,5,1	9,5,7,3,1	9,3,1,7,5

4 Concluding Remarks

Four dimensionality reduction methods are compared for classifying microarrays with the nearest neighbor classifier. Experiments with eight microarray datasets show that dimensionality reduction indeed is effective for nearest neighbor classification.

However, none of the methods used in the study consistently gives the best accuracy on all data sets. Generally, both PCA and PLS results in the highest accuracy for few dimensions whereas RP and IG require more dimensions. Compared to the other three methods, PCA is shown to be more sensitive to the choice of dimensionality, and typically gives poor results in higher dimensions. It can be observed that PLS outperforms the other methods for binary classification problems (Colon, Leukemia, Central Nervous and Prostate).

We have also investigated the accuracy of kNN for different values of k. Generally, k=1 seems to be the best choice for PCA and PLS, while higher values are required for IG.

There are a number of issues that need further exploration. First, additional binary microarray classification tasks could be investigated to test the finding that PLS appears to be superior in these cases. Second, further characterizations of the situations in which the different dimensionality reduction methods are successful could be identified. Furthermore, the possibility of combining several reduced features sets generated by different reduction methods could also be investigated.

Acknowledgements

Financial support from SIDA/SAREC for the first author is greatly acknowledged. The second author was supported by the Information Fusion Research Program (www.infofusion.se) at the University of Skövde, Sweden, in partnership with the Swedish Knowledge Foundation under grant 2003/0104.

References

1. Quackenbush, J.: Microarray analysis and tumor classification. The New England Journal of Medicine 354(23), 2463–2472 (2006)
2. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., Lander, E.S., Loda, M., Kantoff, P.W., Golub, T.R., Sellers, W.R.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1, 203–209 (2002)
3. Kahn, J., Wei, J.S., Ringnér, M., Saal, L.H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C., Peterson, C., Meltzer, P.: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nature Medicine 7, 673–679 (2001)
4. Aha, D.W., Kiblear, D., Albert, M.K.: Instance based learning algorithm. Machine Learning 6, 37–66 (1991)
5. Deegalla, S., Bostrom, H.: Reducing high-dimensional data by principal component analysis vs. random projection for nearest neighbor classification. In: ICMLA 2006. Proceedings of the 5th International Conference on Machine Learning and Applications, pp. 245–250. IEEE Computer Society, Washington, DC, USA (2006)
6. Shlens, J.: A tutorial on principal component analysis, http://www.snl.salk.edu/~shlens/pub/notes/pca.pdf

7. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: KDD 2001. Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 245–250 (2001)
8. Fradkin, D., Madigan, D.: Experiments with random projections for machine learning. In: KDD 2003. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 517–522 (2003)
9. Dasgupta, S., Gupta, A.: An elementary proof of the Johnson-Lindenstrauss lemma. Technical Report TR-99-006, International Computer Science Institute, Berkeley, California, USA (1999)
10. Achlioptas, D.: Database-friendly random projections. In: ACM Symposium on the Principles of Database Systems, pp. 274–281 (2001)
11. Abdi, H.: Partial least squares (pls) regression (2003)
12. de Jong, S.: SIMPLS: An alternative approach to partial least squares regression. Chemometrics and Intelligent Laboratory Systems (1993)
13. StatSoft Inc.: Electronic statistics textbook (2006), http://www.statsoft.com/textbook/stathome.html
14. Boulesteix, A.L.: Pls dimension reduction for classification with microarray data. Statistical Applications in Genetics and Molecular Biology (2004)
15. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
16. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In: Proc. Natl. Acad. Sci., vol. 96, pp. 6745–6750 (1999)
17. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
18. Pomeroy, S.L., Tamayo, P., Gassenbeek, M., Sturla, L.M., Angelo, M., McLaughlin, M.E., Kim, J.Y., Goumnerova, L.C., Black, P.M., Lau, C., Allen, J.C., Zagzag, D., Olson, J.M., Curran, T., Wetmore, C., Biegel, J.A., Poggio, T., Mukherjee, S., Rifkin, R., Califano, A., Stolovitzky, G., Louis, D.N., Mesirov, J.P., Lander, E.S., Golub, T.R.: Prediction of central nervous system embryonal tumour outcome based on gene expression. Nature 415, 436–442 (2002)
19. Alizadeh, A.A., Eisen, M.B., Davis, R.E., Ma, C., Lossos, I.S., Rosenwald, A., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., Powell, J.I., Yang, L., Marti, G.E., Moore, T., Hudson Jr, J., Lu, L., Lewis, D.B., Tibshirani, R., Sherlock, G., Chan, W.C., Greiner, T.C., Weisenburger, D.D., Armitage, J.O., Warnke, R., Levy, R., Wilson, W., Grever, M.R., Byrd, J.C., Botstein, D., Brown, P.O., Staudt, L.M.: Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
20. Ross, D.T., Scherf, U., Eisen, M.B., Perou, C.M., Rees, C., Spellman, P., Iyer, V., Jeffrey, S.S., de Rijn, M.V., Waltham, M., Pergamenschikov, A., Lee, J.C, Lashkari, D., Shalon, D., Myers, T.G., Weinstein, J.N., Botstein, D., Brown, P.O.: Systematic variation in gene expression patterns in human cancer cell lines. Nature Genetics 24(3), 227–235 (2000)
21. Kent Ridge Bio-medical Data Set Repository, http://sdmc.lit.org.sg/GEDatasets/Datasets.html

22. Díaz-Uriarte, R., de Andrés, S.A.: Gene selection and classification of microarray data using random forest. Bioinformatics 7(3) (2006), http://ligarto.org/rdiaz/Papers/rfVS/randomForestVarSel.html
23. Melssen, W., Wehrens, R., Buydens, L.: Supervised kohonen networks for classification problems. Chemometrics and Intelligent Laboratory Systems 83, 99–113 (2006)
24. Melssen, W., Üstün, B., Buydens, L.: Sompls: a supervised self-organising map - partial least squares algorithm. Chemometrics and Intelligent Laboratory Systems 86(1), 102–120 (2006)

Protein Data Condensation for Effective Quaternary Structure Classification

Fabrizio Angiulli[1], Valeria Fionda[2], and Simona E. Rombo[1]

[1] DEIS - Università della Calabria
Via P. Bucci 41C
87036 Rende (CS), Italy
{f.angiulli,simona.rombo}@deis.unical.it
[2] Dept. of Mathematics
Via P. Bucci 31B
87036 Rende (CS), Italy
fionda@si.deis.unical.it

Abstract. Many proteins are composed of two or more subunits, each associated with different polypeptide chains. The number and the arrangement of subunits forming a protein are referred to as *quaternary structure*. The quaternary structure of a protein is important, since it characterizes the biological function of the protein when it is involved in specific biological processes. Unfortunately, quaternary structures are not trivially deducible from protein amino acid sequences. In this work, we propose a protein quaternary structure classification method exploiting the functional domain composition of proteins. It is based on a nearest neighbor condensation technique in order to reduce both the portion of dataset to be stored and the number of comparisons to carry out. Our approach seems to be promising, in that it guarantees an high classification accuracy, even though it does not require the entire dataset to be analyzed. Indeed, experimental evaluations show that the method here proposed selects a small dataset portion for the classification (of the order of the 6.43%) and that it is very accurate (97.74%).

1 Introduction

Proteins are important components of the cell life, catalyzing most of the living cells reactions and controlling virtually all the cellular processes. Proteins are complex molecules composed by individual units called *amino acids*.

While an increasing number of amino acid sequences is produced and stored in public databases, the geometric conformation of a protein can be determined by slow and expensive methods (such as, for example, crystallography and NMR spectrometry). Thus, a number of computational methods have been developed in the last few years to predict and classify protein secondary, tertiary and quaternary structures [4,10,14,15,21]. In this paper, we focuses on protein quaternary structures. In particular, we deal with the problem of efficiently exploiting available databases of amino acid sequences in order to predict the number of subunits of a given protein.

In the rest of this section, we first recall basic concepts concerning protein structure and then point out our contributions.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 810–820, 2007.

Background on protein structure. The sequence of the amino acids of a specific protein is determined by the sequence of the bases in the gene encoding for that protein. Chemical properties of the amino acids composing a protein determine its biological activity. Considering also the order in which amino acid residues lie in the chain, the amino acid sequence is called *primary structure* of the protein. It is possible to univocally identify a protein when the corresponding primary structure is known, but other information about protein spatial conformation are necessary to have a complete description of its biological functions. In particular, the *secondary structure* of a protein is represented by the sequence of the repeated patterns of its backbone folding, without any regard to the conformations of its sidechains; the most common secondary structure configurations are α-helix and β-strand [13]. The *tertiary structure* of a protein refers to the three-dimensional structure of the entire polypeptide, specified by the coordinates of each of its amino acids. Thus, both secondary and tertiary structures contribute to the geometrical description of proteins. Actually, several proteins (e.g., Hemoglobin) are a combination of two or more individual polypeptide chains. The arrangement according to which such subunits assemble is called the protein *quaternary structure*. Quaternary structure refers to the number of subunits involved in forming a protein, to their interconnections and to their organization [12,17]. Biological processes are often influenced by the quaternary structure of proteins involved therein; e.g., the subunit construction of many enzymes provides the structural basis for the regulation of their activities. Proteins having a quaternary structure are called *oligomers*, and may be further classified as *homo-oligomers*, consisting of identical subunits, and *hetero-oligomers*, made of subunits that are different from one another. Furthermore, based on the number of subunits linked together, an homo-oligomer may be an *homodimer*, an *homotrimer*, an *homotetramer*, etc. Proteins consisting of only one subunit are called *monomers*.

Contributions. In this paper, we propose a classification method to individuate the number of subunits of each protein of a given dataset.

To this aim, we exploit protein functional domain information, as already successfully done in previous literature [21]. We encode each protein by a vector whose elements are associated to PFam domains [3]. The number of subunits included in a given protein is then obtained by assigning that protein to a class (e.g., monomers, homodimers, etc.), on the basis of a previously classified dataset and of a suitable classification method.

A few approaches have been recently introduced to support protein quaternary structure classification [5,10,16,21,22]. The most successful of them [21,22] reach at most the 87.5% of overall accuracy, and the maximum dataset size they considered is of about $10{,}000$ proteins. Furthermore, most of the quaternary structure classification methods proposed in the literature store and use the overall dataset, comparing each protein to be classified to each stored protein. This may result hard when large datasets are to be considered.

Our approach gives a contribution in the direction of reducing both the portion of dataset that is necessary to store and, consequently, the number of comparisons to carry out at classification time, allowing sensible space and time savings, while achieving very good accuracy figures.

In particular, we exploit nearest neighbor condensation techniques (as the one recently introduced in [1]) to replace the whole protein dataset with a notable subset that can be then used for the sake of fast protein quaternary structure prediction. With this aim, we use a training set consistent subset for the nearest neighbor decision rule as reference dataset during classification. Let T be a dataset. Having fixed a meaningful distance metrics, a subset S of T is a training set consistent subset of T for the nearest neighbor rule, if S correctly classifies all the objects of T by means of the nearest neighbor rule.To evaluate our method, we considered two different kind of tests. First, we performed the 10-fold cross-validation on a very large protein dataset including $20{,}068$ proteins taken from the SWISSPROT [2] database. The results confirmed the effectiveness of our approach. In fact, we scored an overall accuracy of 97.74%, by using only the 6.51% of the total dataset. This result is important, since pinpoints that our method can be adopted to correctly classify proteins whose quaternary structures are unknown, notably reducing the portion of dataset to analyze. Such a reduction is particularly attractive in the case of protein quaternary structures classification, where large datasets are often to be considered. The second kind of tests concerns the exploitation of the jackknife cross-validation on a non-redundant dataset already used to test another successful technique proposed in the literature [21]. Also in this case, the results we obtained show that our method is more powerful than the previous ones, being able to obtain comparable accuracy in the classification of quaternary structures, even if using only the 45.39% of the whole dataset.

The rest of the paper is organized as follows. Section 2 briefly surveys related work, addresses differences with our approach, and precisely highlights contribution of this work. Section 3 describes our protein quaternary structure classification method. Section 4 presents experimental results. Finally, Section 5 reports conclusions and future work.

2 Related Work

Recently, some techniques have been proposed for protein quaternary structure classification [5,10,16,21,22]. Most of them aim at classifying homo-oligomeric proteins, by storing and exploiting the overall considered dataset and comparing each protein to be classified to each stored protein.

In [10] a software, called Quaternary Structure Explorer (QSE), is presented. The method is based on the analysis of protein amino acid sequences to distinguish between homodimers and non-homodimers, by exploiting rule-based classifiers. The $C4.5$ machine-learning algorithm is used, and proteins are represented by 401 amino acid indices. The overall classification accuracy of the approach is 70%.

The method proposed in [16] uses the function of degree of disagreement (FDOD), that is, a measure of information discrepancy, to discriminate between homodimers and other homooligomeric proteins by exploiting information on the primary structure of proteins, enriched with subsequence distributions of such primary sequences to take into account the effect of residue order on protein structure. The technique scores an overal success rate of the 82.5%.

In [22] both the support vector machine (SVM) and the covariant discriminant algorithms are exploited to predict quaternary structure properties from the protein primary

sequences. The authors consider the amino acid composition and the autocorrelation functions based on the amino acid index profile of the primary sequence, obtaining at the best an accuracy equal to the 87.5%.

The techniques presented in [10,16,22] are able to distinguish just between two classes, that are, homodimers and non-homodimers, whereas our approach is able to discriminate among any number of classes. In this respect, our method is more similar to the two approaches illustrated in the following.

The main contribution of the work [5] is the exploitation of a pseudo amino acid composition to express a protein in terms of a set of discrete numbers, which is more powerful than the standard amino acid composition. In fact, it is able to take into account a considerable amount of sequence-order effects. An overall success rate of 80.1% on a set of 332 test proteins was obtained.

The four approaches described above exploit only protein sequence information, without any regard for protein domain composition. Our method is different, as we consider the protein domain composition that, according also to other studies [19,11,6,4,20], is able to deliver important information about protein structures and functions, which may be related to protein quaternary structure composition.

The approach presented in [21] is based on the functional domain composition of proteins and on the nearest neighbor algorithm (NNA). They performed the jackknife cross-validation test on a non-redundant dataset of 717 proteins, and then exploited the non-redundant dataset to classify an independent dataset of $9{,}951$ proteins. They obtained, resp., an overall success rate of 75.17% and 84.11% for the two dataset.

In our experiments, we used the same non-redundant dataset exploited in [21], enriched in the number of considered domains, obtaining some accuracy improvements (see Section 4). Differently from [21], where a *generalized distance* (which is not a metric) has been exploited in the classification method, we used the Jaccard distance as the distance metric. Furthermore, our technique is more efficient than the one presented in [21] and, in general, than the other related techniques, due to its ability of classifying proteins without the necessity of making comparisons with all the elements of the dataset. Indeed, we are able to extract a relatively small subset of the training set to carry out such a classification without any significant loose in precision.

To summarize, our approach is more general than previous methods, such as [10,16,22], that are specific for the classification of only two classes of protein quaternary structures. Furthermore, we exploited the protein representation which is shown to be the most complete in terms of protein functional information (i.e., functional domain composition), and we achieve high accuracy values even if exploiting small dataset portions. All these features grant to our method highest overall success rate than the other ones presented in the literature (97.74%), making it attractive especially when large protein datasets are to be handled.

3 Classification Method

Next, we describe the classification method to individuate the number of subunits of each protein of a given dataset. In the following we will refer as PQSC-FCNN, for Protein Quaternary Structure Classification through FCNN rule, to the classification method here

presented. In order to design an effective and efficient classification method, different issues are to be addressed. Among them: the feature space and distance metrics to adopt, the classification algorithm, and the suitability of the overall method.

As already pointed out, most of the quaternary structure classification methods proposed in the literature, store and use the whole available dataset, comparing each protein to be classified to each stored protein. This may result hard when large datasets are considered. Hence, we would like to drastically reduce the portion of the dataset that is necessary to store and, consequently, the number of comparisons to carry out, allowing sensible space and time savings.

To this end, we exploit protein functional domain information, and encode each protein by a binary vector whose elements are associated to PFam domains [3]. We adopt the Jaccard metric as our distance measure and exploit the k *nearest neighbor rule* [7], one of the most extensively used nonparametric classification algorithms, which is simple to implement and yet powerful. The rationale underlying this choice is that for this classification rule there exist efficient techniques to reduce both space and time requirements, that we will use in order to reduce the already classified protein dataset.

In the following, the adopted protein representation, distance metrics, classification rule, and data reduction method are detailed.

Protein representation. To characterize proteins, we adopted the functional domain composition, since this kind of representation has been proved to be successful both for the specific problem we analyzed [21], and for the solution of other related problems, such as the prediction of protein-protein interactions [19,11], of protein structures [6] and of protein functions [4,20]. Protein functional domains are elements of the protein structure that are self-stabilizing and often fold independently of the rest of the protein chain. According to the functional domain composition, a protein is represented by a binary vector with size equal to the number of exploited domains. In particular, let D be an ordered set of protein domains, which have been considered to characterize the proteins in a dataset P. Then, each protein $p \in P$ is represented by a vector v_p of $|D|$ elements. The element $v_p[i]$ is set to be one if p contains the *i-th* domain in D, zero otherwise.

Distance metrics. We used the Jaccard metric as our distance metric, which is very suitable for binary data. In particular, the Jaccard distance between two protein vectors v_{p1} and v_{p2} is defined as:

$$d(v_{p1}, v_{p2}) = \frac{n_2 + n_3}{n_1 + n_2 + n_3}$$

where n_1 is the number of indexes for which the correspondent elements are equal to one in both vectors, n_2 is the number of indexes for which the correspondent elements are equal to one in v_{p1} and to zero in v_{p2}, n_3 is the number of indexes for which the correspondent elements are equal to zero in v_{p1} and to one in v_{p2}.

Classification rule. The *nearest neighbor rule* [7] is widely used as a classification algorithm. It is simple to implement and yet powerful, due to its theoretical properties guaranteeing that for all distributions its probability of error is bounded above by twice the Bayes probability of error. The nearest neighbor decision rule can be generalized to the case in which the k nearest neighbors are taken into account. In such a case, a new object is assigned to the class with the most members present among the k nearest

neighbors of the object in the training set. This rule has the additional property that it provides a good estimate of the Bayes error and that its probability of error asymptotically approaches the Bayes error [9].

The naive implementation of the NN rule has no learning phase, since it requires to store all the previously classified data, and then to compare each sample point to be classified to each stored point. In order to reduce both space and time requirements, several techniques to reduce the size of the stored data for the NN rule have been proposed (see [18] for a survey). In particular, among those techniques, the *training set consistent* ones, aim at selecting a subset of the training set that correctly classifies the remaining data through the NN rule.

Data reduction. In order to reduce the reference protein quaternary structure dataset used during classification, we exploited the Fast Condensed Nearest Neighbor rule [1], FCNN for short, an algorithm computing a training set consistent subset for the NN rule. Informally, having fixed a meaningful distance metrics and a dataset T, a subset S of T is a training set consistent subset of T for the nearest neighbor rule, if S correctly classifies all the objects of T by means of the nearest neighbor rule. Thus, loosely speaking, the objects of the subset S can be regarded as representing the objects of T which are not in S, and training set consistent subset methods for the nearest neighbor rule can be regarded as methods to filter out dataset instances which can be considered unessential to correctly classify new incoming objects.

The method is recalled next. We provide some definitions first. We define T as a labelled training set from a metric space with distance metrics d. Let x be an element of T. Then we denote by $nn_k(x,T)$ the kth nearest neighbor of x in T, and by $nns_k(x,T)$ the set $\{nn_i(x,T) \mid 1 \leq i \leq k\}$. $l(x)$ will be the label associated to x. Given a point y, the k-NN rule $\mathrm{NN}_k(y,T)$ assigns to y the label of the class with the most members present in $nns_k(y,T)$. A subset S of T is said to be a *k-training set consistent subset of* T if, for each $y \in (T-S)$, $l(y) = \mathrm{NN}_k(y,S)$. Let S be a subset of T, and let y be an element of S. By $Vor(y,S,T)$ we denote the set $\{x \in T \mid \forall y' \in S, \mathrm{d}(y,x) \leq \mathrm{d}(y',x)\}$, that is the set of the elements of T that are closer to y than to any other element y' of S, called the *Voronoi cell* of y in T w.r.t. S. Furthermore, by $Voren(y,S,T)$ we denote the set $\{x \in (Vor(y,S,T) - \{y\}) \mid l(x) \neq \mathrm{NN}_k(x,S)\}$, whose elements are called *Voronoi enemies* of y in T w.r.t. S. $Centroids(T)$ is the set containing the centroids of each class label in T. The FCNN rule relies on the following property: a set S is a training set consistent subset of T for the nearest neighbor rule if for each element y of S, $Voren(y,S,T)$ is empty.

The FCNN algorithm initializes the consistent subset S with a seed element from each class label of the training set T. In particular, the seeds employed are the centroids of the classes in T. The algorithm is incremental. During each iteration the set S is augmented until the stop condition, given by the property above, is reached. For each element of S, a *representative* element of $Voren(y,S,T)$ w.r.t. y is selected and inserted into S. Such a representative element it is the nearest neighbor of y in $Voren(y,S,T)$, that is, the element $nn(y,Voren(y,S,T))$ of T.

As for the time complexity of the method, let N denote the size of the training set T and let n denote the size of the computed consistent subset S. Then the FCNN rule requires Nn distance computations to compare the elements of T with the elements of S.

Table 1. Overall success rate obtained by the *PQSC-FCNN* method for $k \in \{2, 3, 4\}$ on a dataset of 20,068 proteins

Large dataset						
Classes	**PQSC-FCNN**, $k = 2$		**PQSC-FCNN**, $k = 3$		**PQSC-FCNN**, $k = 4$	
	Corr/Tot	**% Accuracy**	**Corr/Tot**	**% Accuracy**	**Corr/Tot**	**% Accuracy**
Monomer	6,114/6,184	99.45%	6,130/6,184	99.13%	6,135/6,184	99.21%
Homodimer	8,408/8,690	96.75%	8,427/8,690	96.97%	8,402/8,690	96.68%
Homotrimer	1,154/1,190	96.97%	1,150/1,190	96.64%	1,136/1,190	95.46%
Homotetramer	2,422/2,513	96.38%	2,452/2,513	97.57%	2,380/2,513	94.71%
Homopentamer	232/237	97.89%	232/237	97.89%	232/237	97.89%
Homohexamer	759/784	96.81%	761/784	97.07%	742/784	94.64%
Homoheptamer	4/5	80.00%	4/5	80.00%	4/5	80.00%
Homooctamer	457/465	98.28%	458/465	98.49%	458/465	98.49%
Overall	97.60%		97.74%		97.11%	
% Dataset Exploitation	6.43%		6.51%		6.70%	

However, if the distance employed is a metric, a technique exploiting the triangle inequality further reduces this worst case computational cost [1].

4 Experiments

In this section, we illustrate the experimental evaluation of the method proposed in this paper. To build our datasets, we downloaded proteins from the SWISSPROT database [2], available at `www.ebi.ac.uk/swissprot/`, and domains from the PFam database [3], available at `http://www.sanger.ac.uk/Software/Pfam/`.

We considered two different experiments. The first experiment consisted in running the 10-fold cross-validation on a very large protein dataset consisting of 20,068 proteins. The number of considered domains is 1,816. The results of this experiment are shown in Table 1. The first column of the table contains the homo-oligomeric class names, the second, third and fourth ones report both the number of correctly predicted proteins w.r.t. their total number and the percentage of accuracy scored by *PQSC-FCNN* for $k = 2$, $k = 3$ and $k = 4$, respectively, for each class. In the last two rows of the table, the overall accuracy and the percentage of exploited dataset are reported. The results confirmed the effectiveness of our approach. In fact, the maximum overall success rate obtained on the entire dataset is of the 97.74%, and the minimum dataset exploitation is drastically reduced to the 6.43% of the original dataset. In general, as for the classification accuracy the three values of k were comparable, being equivalent on the homopentamers and on the homoheptamers, while only for $k = 3$ and for $k = 4$ the method returns the same results for homooctamers.

Table 2 shows detailed information about the condensed set generated by the method on the overall dataset of 20,068 proteins. In particular, for each class, both the number of elements of the condensed set belonging to that class, and the reduction percentage w.r.t. the total number of elements in that class, are reported. The number of elements and the reduction percentage of the overall condensed set are shown on the last row of the table. By using these three values of k, the method extracted condensed sets with the same size per class for homopentamers, homoheptamers and homooctamers. For the homoheptamer class, the reduction percentage is notably higher than for the other

Table 2. Condensed sets detailed information for the dataset of 20,068 proteins

Condensed set						
Classes	**PQSC-FCNN**, $k = 2$		**PQSC-FCNN**, $k = 3$		**PQSC-FCNN**, $k = 4$	
	Number of elements	**Percentage**	**Number of elements**	**Percentage**	**Number of elements**	**Percentage**
Monomer	98/6,184	1.58%	153/6,184	2.47%	157/6,184	2.54%
Homodimer	643/8,690	7.40%	649/8,690	7.47%	718/8,690	8.26%
Homotrimer	145/1,190	12.18%	101/1,190	8.49%	108/1,190	9.08%
Homotetramer	197/2,513	7.84%	199/2,513	7.92%	157/2,513	6.25%
Homopentamer	17/237	7.17%	17/237	7.17%	17/237	7.17%
Homohexamer	74/784	9.44%	74/784	9.44%	79/784	10.08%
Homoheptamer	3/5	60.00%	3/5	60.00%	3/5	60.00%
Homooctamer	29/465	6.24%	29/465	6.24%	29/465	6.24%
Overall	1,206	6,01%	1,225	6.10%	1,268	6.32%

classes, due to the few elements belonging to that class (only 5 ones). The reduction percentage on the overall dataset is 6.01% for $k = 2$, 6.10% for $k = 3$ and 6.32% for $k = 4$. This show the power of the method, as it is sufficient to explore only a bit more than the 6% of the overall dataset to (most probably) classify a new protein.

Both the exploited protein datasets and the correspondent domains can be downloaded at the web-links reported in Section 5.

In order to compare our method with a related one, in the second kind of experiments we considered the non-redundant protein dataset discussed by Yu et al. in [21]. The main goal of this comparison is to show that our method may have accuracy comparable to those of related methods, while sensibly reducing the amount of labeled data to exploit during classification. In particular, we point out that the method presented in [21] utilizes a non redundant version of the overall protein dataset in order to cope with problems associated with management of large data sets. As we will show in the following, our method will be able to halve even this non redundant dataset, while maintaining the same accuracy as the competitor method.

Yu et al. adopted an approach based on the functional domain composition and employed the nearest neighbor algorithm (NNA) to classify protein quaternary structures. They represented the 717 considered proteins by 540 domains. Here, we enlarge the number of considered domains to 1,253 in order to obtain a more accurate representation of data. Thus, we compare *PQSC-FCNN* with *NNA* by running the jackknife cross-validation on the non-redundant dataset, by considering the same 1,253 domains representation for all methods.

We run *PQSC-FCNN* exploiting the Jaccard metric, whereas *NNA* has been run with the generalized distance exploited in [21]. The results are illustrated in Table 3. The first column of the table contains the homo-oligomeric classes, the second, third, fourth and fifth ones illustrate both the number of correctly predicted objects w.r.t. the total number of them and the percentage of accuracy scored by *PQSC-FCNN* for $k = 2$, for $k = 3$ and for $k = 4$, and *NNA*, respectively, for each considered class. In the last two rows of the table, the overall accuracy and the percentage of exploited dataset for each method are reported. We can observe that all the considered techniques return the same results for the two classes of homopentamers and homooctamers. The only class for which *PQSC-FCNN* does not obtain higher accuracy than the *NNA* is the homodimer class. However, for $k = 4$ it obtains a success rate of 78.51% for that class, w.r.t. the 80.00% scored by the *NNA*, and thus, also in this case, the two methods achieve comparable results.

Table 3. Comparison of overall success rate obtained by PQSC-FCNN and NNA

Non-redundant dataset								
Classes	**PQSC-FCNN**, $k = 2$		**PQSC-FCNN**, $k = 3$		**PQSC-FCNN**, $k = 4$		**NNA**	
	Corr/Tot	**% Accuracy**	**Corr/Tot**	**% Accuracy**	**Corr/Tot**	**% Accuracy**	**Corr/Tot**	**% Accuracy**
Monomer	177/208	85.10%	174/208	83.65%	178/208	85.58%	168/208	80.77%
Homodimer	250/335	74.63%	250/335	74.63%	263/335	78.51%	268/335	80.00%
Homotrimer	28/40	70.00%	27/40	67.50%	27/40	67.50%	28/40	70.00%
Homotetramer	53/95	55.79%	52/95	54.74%	42/95	44.21%	53/95	55.79%
Homopentamer	11/11	100.00%	11/11	100.00%	11/11	100.00%	11/11	100.00%
Homohexamer	9/23	39.13%	9/23	39.13%	10/23	43.48%	7/23	30.43%
Homooctamer	2/5	40.00%	2/5	40.00%	2/5	40.00%	2/5	40.00%
Overall accuracy	73.92%		73.22%		74.34%		74.90%	
% Dataset Exploitation	46.79%		47.35%		45.39%		100%	

For the homotrimer and the homotetramer classes, the *PQSC-FCNN* for $k = 2$ and the *NNA* have the same accuracy scores, whereas in the remaining cases (i.e., monomers and homoexamers), *PQSC-FCNN* always scores better accuracy than *NNA*, also with some sensible improvements. In particular, for $k = 4$, *PQSC-FCNN* scores the accuracy value of 85.58% for monomers, which is about 4.81% higher than the success rate obtained by the *NNA*, whereas it scores the accuracy value of 43.48% for homohexamers, which is about 13.05% higher than the success rate obtained by the *NNA* for the same class and represents the best accuracy improvement w.r.t. *NNA* we obtained.

These results are significant since, for monomers and homoexamers, our method has been able to obtain more accurate results than *NNA* even exploiting only the 45.39% of the overall dataset, whereas [21] did not make any dataset reduction.

Summarizing, the *PQSC-FCNN* method is able to reach an overall success rate that is greater than the *NNA*, even if it exploits only the $45.39 - 47.35\%$ of the original dataset. This means that *PQSC-FCNN* is more efficient than the method [21], allowing both time and space savings without any significant loss in accuracy but, rather, often allowing success rate improvements.

5 Concluding Remarks

We proposed a classification method for protein quaternary structures, by exploiting protein functional domain information and FCNN rule. Experimental evaluations showed that our approach is able to reduce both the portion of protein dataset that is necessary to store and the number of comparisons to carry out, allowing sensible space and time savings even if guaranteeing high values of accuracy. Thus, the method is competitive w.r.t. other methods presented in the literature. We illustrated some tests we carried out on homo-oligomeric proteins, confirming the validity of our approach. Future efforts will be devoted to investigate the problem of quaternary structure prediction for hetero-oligomers.

Additional material. The protein datasets we exploited are available on-line. In particular, the non-redundant dataset of 717 proteins is listed at:

`http://siloe.deis.unical.it/PQSC_FCNN/non_redundant_dataset.pdf`,

together with the corresponding 1,253 domains we exploited for the functional domain composition, while the 20,068 proteins of the second dataset and their 1,816 are available at:

`http://siloe.deis.unical.it/PQSC_FCNN/20068_dataset.pdf`.

References

1. Angiulli, F.: Fast condensend nearest neighbor rule. In: Proc. of the 22nd International Conference on Machine Learning, Bonn, Germany (2005)
2. Bairoch, A., Apweiler, R.: The swiss-prot protein sequence data bank and its new supplement trembl. Nucleic Acids Research 24(1), 21–25 (1996)
3. Bateman, A., Birney, E., Cerruti, L., Durbin, R., Etwiller, L., Eddy, S.R., Griffiths-Jones, S., Howe, K.L., Marshall, M., Sonnhammer, E.L.L.: The pfam protein families database. Nucleic Acids Reserch 30(1), 276–280 (2002)
4. Cai, Y.D., Doig, A.J.: Prediction of saccharomyces cerevisiae protein functional class from functional domain composition. Bioinformatics 20(8), 1292–1300 (2004)
5. Chou, K.C., Cai, Y.D.: Predicting protein quaternary structure by pseudo amino acid composition. Proteins: Structure, Function, and Genetics 53(2), 282–289 (2003)
6. Chou, K.C., Cai, Y.D.: Predicting protein structural class by functional domain composition. Biochemical and biophysical research communications 321(4), 1007–1009 (2004)
7. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Trans. on Inform. Th. 13(1), 21–27 (1967)
8. Devroye, L., Gyorfy, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition. Springer, New York (1996)
9. Fukunaga, K., Hostetler, L.D.: k-nearest-neighbor bayes-risk estimation. IEEE Transactions on Information Theory 21, 285–293 (1975)
10. Garian, R.: Prediction of quaternary structure from primary structure. Bioinformatics 17(6), 551–556 (2000)
11. Kim, W.K., Park, J., Suh, J.K.: Large scale statistical prediction of protein-protein interaction by potentially interacting domain (pid) pair. In: Genome informatics. International Conference on Genome Informatics, vol. 13, pp. 42–50 (2002)
12. Klotz, I.M., Langerman, N.R., Darnall, D.W.: Quaternary structure of proteins. Annual review of biochemistry 39, 25–62 (1970)
13. Lesk, A.M.: Introduction to Protein Architecture. Oxford University Press, Oxford (2001)
14. Meiler, J., Baker, D.: Coupled prediction of protein secondary and tertiary structure. Proceedings of the National Academy of Sciences of the United States of America 100(21), 12105–12110 (2003)
15. Pollastri, G., Przybylski, D., Rost, B., Baldi, P.: Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. Proteins 47, 228–235 (2002)
16. Song, J., Tang, H.: Accurate classification of homodimeric vs other homooligomeric proteins using a new measure of information discrepancy. Journal of chemical information and computer sciences 44(4), 1324–1327 (2004)
17. Sund, H., Weber, K.: The quaternary structure of proteins. Angewandte Chemie (International eds in English) 5(2), 231–245 (1966)
18. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning 38(3), 257–286 (2000)
19. Wojcik, J., Schachter, V.: Protein-protein interaction map inference using interacting domain profile pairs. Bioinformatics 17(1), 296–305 (2001)

20. Yu, X., Lin, J., Shi, T., Li, Y.: A novel domain-based method for predicting the functional classes of proteins. Chinese Science Bullettin - English Edition- 49(22), 2379–2384 (2004)
21. Yu, X., Wang, C., Li, Y.: Classification of protein quaternary structure by functional domain composition. BMC Bioinformatics 7(187) (2006)
22. Zhang, S.W., Pan, Q., Zhang, H.C., Zhang, Y.L., Wang, H.Y.: Classification of protein quaternary structure with support vector machine. Bioinformatics 19(18), 2390–2396 (2003)

PINCoC: A Co-clustering Based Approach to Analyze Protein-Protein Interaction Networks

Clara Pizzuti[1] and Simona E. Rombo[2]

[1] ICAR-CNR, Via P. Bucci 41C, 87036 Rende (CS), Italy,
pizzuti@icar.cnr.it
[2] DEIS - Università della Calabria, Via P. Bucci 41C, 87036 Rende (CS), Italy,
simona.rombo@deis.unical.it

Abstract. A novel technique to search for functional modules in a protein-protein interaction network is presented. The network is represented by the adjacency matrix associated with the undirected graph modelling it. The algorithm introduces the concept of *quality* of a sub-matrix of the adjacency matrix, and applies a greedy search technique for finding local optimal solutions made of dense sub-matrices containing the maximum number of ones. An initial random solution, constituted by a single protein, is evolved to search for a locally optimal solution by adding/removing connected proteins that best contribute to improve the *quality* function. Experimental evaluations carried out on *Saccaromyces Cerevisiae* proteins show that the algorithm is able to efficiently isolate groups of biologically meaningful proteins corresponding to the most compact sets of interactions.

1 Introduction

One of the most important challenges of the post-genomic era is the analysis of the complex biological processes in which proteins are involved. Recently, great attention has been addressed to the whole set of protein interactions of a given organism, known as *interactome* or *protein-protein interaction (PPI) network*. Many studies have been driven to predict and understand functional properties of proteins starting from interactomes (e.g., [6,5,12]). In the last few years, a vast amount of new protein interactions have been discovered and made available. This has spurred the search for automated and accurate tools to identify significant parts of this data.

PPI networks are often modelled as graphs where nodes represent proteins and edges represent pairwise interactions. Many current efforts aim at clustering dense regions of a given PPI network, since it has been observed by biologists that groups of highly interacting proteins could be involved in common biological processes. A number of approaches have been proposed to extract relevant modules from PPI networks [4,5,1,13,12]; some of them rely on traditional hierarchical clustering methods [7], other ones are based on graph partitioning algorithms [3,11,8]. The obtained results have been found to strongly depend on the adopted approach, and on the input parameters fixed by the user. Most methods, in fact, require the number of clusters to be known in advance. However, this information is not always available, thus some algorithms are executed with different cluster numbers and results satisfying a quality criteria are considered to be the most reliable. Obviously, the necessity of running an algorithm different times may cause

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 821–830, 2007.

losses in efficiency. Another problem that arises in PPI networks is the choice of the metric adopted to measure the distance between two proteins. In this kind of graphs, due to the structure of the interactions, it has been found that the distances among many nodes are often identical. In such a case the adopted clustering method fails in finding good solutions, due to the presence of ties that have to be solved arbitrarily.

In this paper, we present a novel technique, based on a co-clustering approach [9], to search for functional modules in protein-protein interaction networks. Co-clustering methods, differently from clustering approaches, aim at simultaneously grouping both the dimensions of a data set. We model a protein-protein interaction network by an undirected graph and represent it as the binary adjacency matrix A of this graph, where rows and columns correspond to proteins and a 1 entry at the position (i,j) means that the proteins i and j interact. The *PPI network Co-Clustering* based algorithm, named *PINCoC*, applies a greedy search technique for finding local optimal solutions made of dense sub-matrices containing the maximum number of ones. The notion of *quality* of a sub-matrix is introduced. High quality sub-matrices should correspond to modules of the input interactome having a significant biological function. The algorithm starts with an initial random solution constituted by a single protein and searches for a locally optimal solution by adding/removing connected proteins that best contribute to improve the *quality* function. In order to escape poor local maxima, with a fixed probability, the protein causing the minimal decrease of the *quality* function is removed. When the algorithm cannot improve any more the solution found so far, the computed cluster is returned. To limit the effects of the initial random choice of a protein to build a cluster, one step of backtracking is executed. Each protein belonging to the solution is at turn temporary removed, and eventually substituted with a new one that best improves the *quality* function. At this point a new random protein is chosen, and the process is repeated until all the proteins are assigned to any group. In the hard scenario of strongly connected networks, where the detection of the most functionally related proteins is a difficult task due to the high number of connections, our algorithm is able to efficiently isolate those groups of proteins corresponding to the most compact sets of interactions. In the experimental result section we validate the clusters found by *PINCoC* through the *SGD Gene Ontology Term Finder* and compare our results with other studies made in the literature [1,8]. We show that the obtained clusters are recognized to be biologically meaningful.

The paper is organized as follows. The next section defines the problem of clustering PPI networks and the adopted notation. Section 3 describes the proposed algorithm. Section 4 illustrates the experiments we carried out on a set of S. Cerevisiae proteins and compare the obtained results with those of [1,8]. Finally, in Section 5 we draw our conclusions.

2 Notation and Problem Definition

In this section the notation used in the paper is introduced and the formalization of the problem of clustering PPI networks as a co-clustering problem is provided.

A PPI network $\mathcal{P}$ can be modelled as an undirected graph $G = (V, E)$ where the nodes V correspond to the proteins and the edges E correspond to the pairwise

interactions. If the network is constituted by N proteins, the associated graph can be represented with its $N \times N$ adjacency matrix A, where the entry at position (i, j) is 1 if there is an edge from node i to node j, 0 otherwise. Since the graph G is undirected, the adjacency matrix is symmetric. Note that the mathematical definition of adjacency matrix assumes that the main diagonal contains a 1 value at position (i, i) only if there is a loop at vertex i. In the biological context a protein connected with itself is not meaningful. However, by convention, we assume that the main diagonal of the adjacency matrix A of a PPI network contains all ones. This means that if a row of A is constituted by all zeroes except one position i with value 1, the protein corresponding to node i does not interact with any other protein. The problem of finding dense regions of a PPI network $\mathcal{P}$ can thus be transformed in that of finding dense subgraphs of the graph G associated with $\mathcal{P}$, and consequently, dense sub-matrices of the adjacency matrix A corresponding to G. Searching for dense sub-matrices of a matrix A can be viewed as a special case of co-clustering a binary data matrix where the set of rows and columns represent the same concept. In order to better explain the idea, first a definition of co-clustering is given, and then the formalization of the problem of clustering proteins as a co-clustering problem is provided. Co-clustering [9], also known as bi-clustering, differently from clustering, tries to simultaneously group both the dimensions of a data set. For example, when clustering genes with respect to a set of experimental conditions, not all the genes are relevant for all the experimental conditions, but groups of genes are often co-regulated and co-expressed only under specific conditions. In this application domains the idea of co-clustering both the dimensions turns to be more beneficial and interesting than clustering with only one dimension. Let A be an $N \times M$ data matrix of binary values. Let $X = \{I_1, \ldots, I_N\}$ denote the set of rows of A and $Y = \{J_1, \ldots, J_M\}$ the set of columns of A.

Definition 1. *A co-cluster is a sub-matrix $B = (I, J)$ of A, where I is a subset of the rows X of A, and J is a subset of the columns Y of A.*

The problem of co-clustering can then be formulated as follows: given a data matrix A, find row and column maximal groups which divide the matrix into regions that satisfy some homogeneity characteristics. The kind of homogeneity a co-cluster must fulfil depends on the application domain. In our case we would like to find as many proteins as possible having the highest number of interactions. This corresponds to identify highly dense squared sub-matrices, i.e. containing as many 1 values as possible. Higher the number of ones, more likely those proteins are to be functionally related. In the following we introduce a *quality* function that tries to obtain both these objectives. Note that the adjacency matrix A associated with a PPI network is a squared matrix of dimension $N \times N$, where N is the number of proteins. This means that any co-cluster $B = (I, J)$ of A has the property that the set I of rows and the set J of columns coincide. In particular, being A symmetric, any co-cluster found is symmetric too.

Let a_{iJ} denote the *mean value* of the ith row of the co-cluster $B = (I, J)$, and a_{Ij} the mean of the jth column of $B = (I, J)$. More formally,

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}, \text{ and } a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$$

The *volume* v_B of a co-cluster $B = (I, J)$ is the number of 1 entries a_{ij} such that $i \in I$ and $j \in J$, that is $v_B = \sum_{i \in I, j \in J} a_{ij}$.

Definition 2. *Given a co-cluster $B = (I, J)$, let a_{iJ} be the mean of the ith row of B, and let a_{Ij} be the mean of the jth column of B. The power mean of B of order r, denoted as $\mathbf{M}_r(B)$ is defined as*

$$\mathbf{M}_r(B) = \frac{\sum_{i \in I}(a_{iJ})^r + \sum_{j \in J}(a_{Ij})^r}{|I| + |J|}$$

Since B is symmetric, $|I| = |J|$ and $a_{iJ} = a_{Ij}$, thus the power mean can be reduced to

$$\mathbf{M}_r(B) = \frac{\sum_{i \in I}(a_{iJ})^r}{|I|}$$

A quality measure based on volume and row/column mean, that allows the detection of maximal and dense sub-matrices, can be defined as follows.

Definition 3. *Given a co-cluster $B = (I, J)$, let $\mathbf{M}_r(B)$ be the power mean of B of order r. The quality of B is defined as $Q(B) = \mathbf{M}_r(B) \times v_B$.*

i.e. the quality of a co-cluster B is the product between the power mean of B of order r, and the number of non-zero entries in B. The quality $Q(B)$ of the co-cluster $B = (I, J)$ is equal to $|I| \cdot |I|$ only when each entry of B is one, thus $Q(B)$ is upper bounded by its volume, i.e. $Q(B) \leq v_B \leq |I| \cdot |I|$. When B contains zero entries, the *quality* is a fraction of v_B. Notice that, adding a row/column composed only by ones or removing a row/column composed only by zeros, always improves the *quality* of the co-cluster.

When $r = 1$ the power mean coincides with the standard mean. However, the mean of a binary matrix of fixed volume (i.e., having the same number of ones), assumes always the same value independently where the 1/0 values are positioned. This means that it is not able to distinguish matrices corresponding to PPI networks having the same total number of interactions but different structure. Consider for example the two sub-matrices and the associated protein graphs showed in Figure 1. The total number of ones, i.e., of interactions, is equal to 13 in both cases, but the way the proteins interact is different. Intuitively, the graph in Figure 1(b) represents a more compact set of interactions than the one in Figure 1(a). If we compute the power mean of order 1 the value is 0.260 for both of the illustrated matrices, whereas the power mean of order 2 is 0.140 for the matrix on the left and 0.148 for the matrix on the right. Since the volume (the number of ones) is 13 for both matrices, the *quality* function in the former case is 3.38 for both the matrices, while for $r = 2$ it is 1.820 for the first matrix and 1.924 for the second one. Thus, $r = 2$ is more suited to characterize different ways in which proteins interact. However, it is worth to point out that increasing the value of r biases the *quality* function towards matrices containing a low number of zeroes but of lower volume. Thus the choice of r should be done by considering the density of the adjacency matrix. In the next section the *PPI network Co-Clustering* based algorithm *PINCoC*, is presented. The method uses the concept of *quality* to find maximally dense regions in the binary data adjacency matrix.

3 Algorithm Description

In this section we present *PINCoC*, an algorithm for clustering a PPI network $\mathcal{P}$ represented through the adjacency matrix A of the graph associated with $\mathcal{P}$. Let $A = (X, Y)$

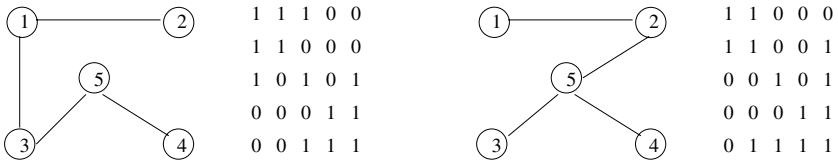


Fig. 1. Matrices with equal mean value but different network structure

be the $N \times N$ adjacency matrix where $X = \{I_1, \ldots, I_N\}$ denote the set of rows of A and $Y = \{J_1, \ldots, J_N\}$ the set of columns of A. Each row/column of A corresponds to a protein, thus in the following we use the two terms as synonyms.

A co-cluster $B = (I, J)$ can be encoded as a binary string b of length N, where N is the number of rows/columns of the adjacency matrix. If the value of the i-th bit is set to 1 it means that the corresponding i-th protein belongs to the co-cluster.

Algorithm *PINCoC*
Input:
- a 0-1 adjacency matrix $A = (X, Y)$ of dimension $N \times N$
- maximum number of iterations allowed max_flips
- probability p of a REMOVE-MIN move (0 = no remove move)
Ouput: a clustering $B = \{B_1, \ldots, B_k\}$ of the PPI network corresponding to A
set $B = \emptyset$, $A_0(X_0, Y_0) = A(X, Y)$, i = 0
while A_i is not the empty matrix
choose at random a row k of A_i, set $I_i = \{k\}$, $J_i = \{k\}$, $B_i = (I_i, J_i)$
set $f = 0$, $quality = -\infty$, $local\_maximum = false$
  while $f < max\_flips$ and not $local\_maximum$
    let $0 \leq \overline{p} \leq 1$ a random generated number
    if $\overline{p} > p$ **then**
      let $\overline{B_i} = (\overline{I_i}, \overline{J_i})$ the co-cluster obtained from B_i
      after the move that maximize the *quality*
      if $Q(\overline{B_i}) > Q(B_i)$ **then**
        accept the move, $I_i = \overline{I_i}$, $J_i = \overline{J_i}$, and update $Q(B_i)$
      else
        set $local\_maximum = true$
    else
      remove the row/column of B_i scoring the minimum decrease of the *quality* function
    $f = f + 1$
  end while
  execute one-pass backtracking
  $B = B \cup B_i$
  set $X_{i+1} = X_i - I_i, Y_{i+1} = Y_i - J_i, A_{i+1} = (X_{i+1}, Y_{i+1})$
end while
  assign the singletons
  return the clustering $B = \{B_1, \ldots, B_k\}$

Fig. 2. The *PINCoC* algorithm

The algorithm, showed in figure 2, receives in input a 0-1 adjacency matrix, the maximum number of times (max_flips) that a flip can be done, and the probability (p) of executing a REMOVE-MIN move (these two latter input parameters are explained shortly). *PINCoC* starts with an initial random co-cluster $B = (I_i, J_i)$ constituted by a single row and a single column such that $I = \{k\}$ and $J = \{k\}$, where $1 \leq k \leq N$ is a

random row/column index. Then it evolves the initial co-cluster by successive transformations of B_i, until the *quality* function is improved. The transformations consist in the change of membership (called $flip$ or $move$) of the row/column that leads to the largest increase of the *quality* function. If a bit is set from 0 to 1 it means that the corresponding protein, which was not included in the co-cluster B_i, is added to B_i. Viceversa, if a bit is set from 1 to 0 it means that the corresponding protein is removed from the co-cluster. During its execution, in order to avoid get trapped into poor local maxima, instead of performing the flip maximizing the $quality$, with a user-provided probability p the algorithm selects the row/column of B_i scoring the minimum decrease of the *quality* function, and removes it from B_i. This kind of flip is called REMOVE-MIN. The flips are repeated until either a preset of maximum number of flips (max_flips) is reached, or the solution cannot ulteriorly be improved (get trapped into a local maximum). Until the stop condition is not reached, it executes a REMOVE-MIN move with probability p, and a greedy move with probability $(1 - p)$. When the inner loop stops, the co-cluster $B_i = (I_i, J_i)$ is returned. At this point the algorithm performs one step of backtracking, i.e. for each $h \in I_i$, it temporary removes h from I_i and tries to find a node l such that $I_i - \{h\} \cup \{l\}$ improves the *quality* of B_i. In such a case h is removed and l is added. If more than one l node exists, the one generating the better improvement of $Q(B_i)$ is chosen. Finally, B_i is added to B, its rows/columns are removed from A, a new random co-cluster is generated, and the process is repeated until all the rows/columns have been assigned. Some of the clusters obtained at the end of the algorithm could be constituted by a single protein because all its neighboring nodes have already been assigned to a group. This situation happens for those proteins that have few interactions and thus they have not been assigned to any group because their contribution was considered marginal. However, we chose to handle such singletons by adopting the following strategy. Let h be a singleton protein, $n_1, \ldots, n_h$ its neighboring proteins, i.e. the proteins having a direct interaction with h, and $B_{n_1}, \ldots B_{n_h}$ their corresponding clusters (note that the B_{n_i} are not necessarily distinct). Then h is assigned to the cluster B_{n_i} s.t. $Q(B_{n_i} \cup \{h\})$ is maximum, i.e. whose *quality* function has the better improvement or the lowest decrease. In the experimental results section we show that *PINCoC* is able to generate clusters of proteins both dense and biologically meaningful. The temporal cost of the algorithm to compute a single cluster $B_i = (I_i, J_i)$ is upper bounded by

$$max\_flips \times C_q \times [(1-p) \times N + pN] + C_q \times \mid I_i \mid \times N = C_q \times N \times (max\_flips + \mid I_i \mid)$$

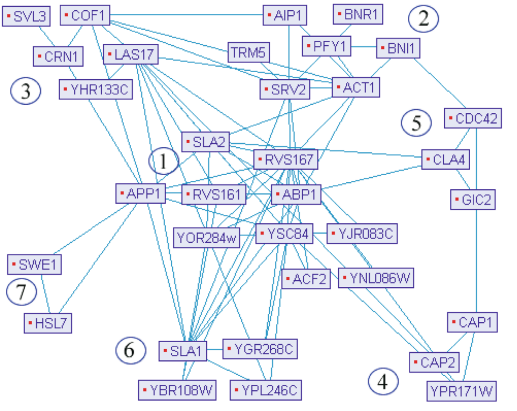
where C_q is the cost of computing the *quality* of the co-cluster after performing a move. In order to reduce the complexity of C_q, we maintain, together with the current co-cluster $B_i = (I_i, J_i)$, the mean values a_{iJ}, for each $i \in I$, and the volume v_{IJ}. Thus, computing the $|I_i|$ mean values a_{iJ} $(1 \leq i \leq |I|)$ after performing a move can be done efficiently in time $|I_i|$, i.e. in time linear in the co-cluster dimensions, by exploiting the values maintained together with the current co-cluster.

4 Experimental Validation

In this section we apply *PINCoC* on a set of 34 proteins coming from the well known *S. cerevisiae* network. This set, extracted from the *DIP* database (*http://dip.doe-mbi*

Clusters	Gene Ontology term	p-value
1	Actyn cytoskeleton organization and biogenesis	$2.25 \cdot 10^{-10}$
2	Rensponse to osmotic stress	$9.63 \cdot 10^{-07}$
3	Actin filament organization	$3.1 \cdot 10^{-04}$
4	Actin cytoskeleton organization and biogenesis	$4.63 \cdot 10^{-06}$
5	Rho protein signal transduction	$1.11 \cdot 10^{-06}$
6	Actin cortical patch assembly	$4.4 \cdot 10^{-04}$
7	G2/M transition of mytotic cell cycle	$5.3 \cdot 10^{-04}$

(a)



Clusters	Proteins
1	sla2 abp1 app1 yor284w rvs161 rvs167 ynl086w yjr083c ysc84 acf2
2	bnr1 pfy1 trm5 bni1 aip1 srv2 act1
3	svl3 crn1 cof1 las17 yhr133c
4	cap1 cap2 ypr171w
5	cdc42 cla4 gic2
6	sla1 ybr108w ypl246c ygr268c
7	swe1 hsl7

(b)

Fig. 3. (a) Clusters validation by Gene Ontology term finder; (b) graphical view of the obtained clusters drawn using PIVOT [10]

.ucla.edu/) has already been well studied and characterized in the literature [6,1]. In the following, we first present the clusters obtained by our method and we validate their biological meaningfulness by using the *SGD Gene Ontology Term Finder* (*http://db .yeastgenome.org/cgi-bin/GO/goTermFinder*). Then we compare our results with those obtained by Arnau and Marìn [1], and King et al. [8] showing that the clustering returned by our method is meaningful and comparable with the other two approaches.

The *PINCoC* algorithm has been implemented in C++, and all the experiments have been performed on a Pentium 4 machine, 1800MHz, 1GB RAM, by using $r = 2$, $max\_flips = 100$, $p = 0.1$. The results obtained are summarized in figure 3. In particular, figure 3(a) shows the table containing the seven clusters returned, the GO term obtained when querying the *SGD Gene Ontology Term Finder* with the proteins belonging to our clusters, and the corresponding p-value. The p-value is a commonly used measure of the statistical and biological significance of a cluster. It gives the probability that a given set of proteins occurs by chance. In particular, given a cluster of size n and m proteins sharing a particular biological annotation, then the probability of observing m or more proteins that are annotated with the same GO term out of those n proteins, according to the Hypergeometric Distribution, is: $p-value = \sum_{i=m}^{n} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{n}}$, where N is the number of proteins in the database with M of them known to have that same annotation [2]. Thus, the closer the p-value to zero, the more significant the associated GO term. In the table we show the smallest p-value found over all the functional groups. We can observe that the p-value of our clusters varies between $2.25 \cdot 10^{-10}$

and $5.3 \cdot 10^{-04}$, values all sufficiently low to consider relevant the biological meaningfulness of the corresponding clusters. Figure 3(b) shows a graphical representation of the interactions among the considered proteins, indicated by the names according to the Gene Ontology notation, and a list of the proteins participating to each cluster. The graph has been drawn using PIVOT [10]. It is worth to point out that the biological significance of the seven clusters agrees with the functional classification reported in [6]. *PINCoC*, in fact, is able to correctly distinguish proteins involved in different processes such as, for example, actin patch assembly and patch mediated endocytosis (Cluster 1), actin-capping proteins (Cluster 4), CDC42 signaling pathway (Cluster 5), control of the morphogenesis checkpoint (Cluster 7).

In order to better assess the quality of the results obtained by *PINCoC*, we now compare them with those obtained in [1] and [8]. Arnau and Marìn [1] proposed the hierarchical clustering method UVCLUSTER, that iteratively explores the distance data sets to analyze protein-protein interaction networks. UVCLUSTER uses an agglomerative hierarchical clustering twice. The first time it considers the *primary distances*, that is, the minimum number of interactions required to connect two proteins, and generates K alternative clustering solutions. The value of K must be given by the user. The second time it clusters again the set of proteins but using the *secondary distances*, defined as the percentage of clusters in which two proteins do not appear together.

The second algorithm we consider for comparison is the Restricted Neighborhood Search Clustering (RNSC), proposed by of King et al. [8]. RNSC is a cost-based local search algorithm that explores the solution space of all the possible clusterings to minimize a cost function that refelcts the number of inter-cluster and intra-cluster edges.

Table 1 reports the clusterings obtained by *PINCoC*, UVCLUSTER and RNSC with the list of proteins for each cluster, the fraction of proteins in each cluster that have been recognized to participate to a specific biological process with the p-value reported in the last column. RNSC needs some input parameters. In our experimentation we used the values reported by Brohèe and van Helden [4], who have extensively analyzed RNSC to determine the best parameter values with respect to (i) the best matching complex found in a cluster, denoted by RNSC$_a$, and (ii) how well a given cluster isolates complexes from other clusters, denoted by RNSC$_s$. Note that the p-values of the clusters reported for UVCLUSTER differ from those appearing in [1] because the authors computed the values on the January 2004 release of the DIP database, containing 4721 proteins. At present DIP contains 5027 proteins. For each cluster found by *PINCoC*, we report the cluster (or the clusters) obtained by UVCLUSTER and RNSC that has the maximum number of common proteins with *PINCoC*. The names of the common proteins with UVCLUSTER are highlighted in bold, those of the common proteins with RNSC are highlighted in italic. The symbol '–' means that no significant ontology term has been found for that cluster. The table points out that our first cluster is bigger than those generated by both UVCLUSTER and RNSC, and has a lower p-value. The second cluster found by *PINCoC* partially includes two different clusters found by UVCLUSTER, and other two different clusters found by RNSC (note that we use RNSCa, s for short when both the two RNSC runs returned the same cluster). Both the two groups generated by UVCLUSTER and those generated by RNSC have higher p-value than the *PINCoC* cluster. In correspondence of the third cluster

Table 1. Clusters validation by Gene Ontology term finder, updated at September 2007, for *PINCoC*, UVCLUSTER [1] and RNSC [8]

Methods	Clusters	Proteins Fraction	p-value
PINCoC	***sla2 abp1 yor284w rvs167 ysc84*** app1 *rvs161* ynl086w *yjr083c* acf2	7/10	$2.25 \cdot 10^{-10}$
UVCluster	**sla2 abp1 yor284w rvs167 ysc84** sla1 ygr268c	6/7	$3.29 \cdot 10^{-09}$
RNSC$_{a,s}$	*sla2 abp1 yor284w rvs167 ysc84 rvs161 yjr083c*	7/7	$7.63 \cdot 10^{-07}$
PINCoC	***bnr1 bni1 pfy1 act1 srv2 aip1*** **trm5**	6/7	$9.63 \cdot 10^{-07}$
UVCluster	**bnr1 bni1 pfy1**	2/3	$3.99 \cdot 10^{-05}$
UVCluster	**act1 srv2 aip1 trm5** cof1	2/5	$3.50 \cdot 10^{-04}$
RNSC$_a$	*bnr1 bni1 pfy1*	2/3	$3.99 \cdot 10^{-05}$
RNSC$_s$	*bnr1 pfy1*	2/2	$3.70 \cdot 10^{-04}$
RNSC$_{a,s}$	*act1 srv2 aip1* cof1	4/4	$1.30 \cdot 10^{-04}$
PINCoC	***svl3 crn1 las17 yhr133c*** cof1	3/5	$3.1 \cdot 10^{-04}$
UVCluster	ypl246c **las17 yhr133c**	–	–
UVCluster	**crn1 svl3**	–	–
RNSC$_{a,s}$	*las17 yhr133c*	–	–
RNSC$_{a,s}$	*crn1 svl3*	–	–
PINCoC	***cap1 cap2 ypr171w***	3/3	$4.63 \cdot 10^{-06}$
UVCluster	**cap1 cap2 ypr171w**	3/3	$4.63 \cdot 10^{-06}$
RNSC$_{a,s}$	*cap1 cap2 ypr171w*	3/3	$4.63 \cdot 10^{-06}$
PINCoC	***cdc42 cla4 gic2***	3/3	$1.11 \cdot 10^{-06}$
UVCluster	**cdc42 cla4 gic2**	3/3	$1.11 \cdot 10^{-06}$
RNSC$_a$	*cdc42 cla4 gic2*	3/3	$1.11 \cdot 10^{-06}$
RNSC$_s$	*cdc42 cla4 gic2* bni1	4/4	$2.83 \cdot 10^{-09}$
PINCoC	*sla1* ***ybr108w*** ypl246c *ygr268c*	3/3	$1.11 \cdot 10^{-06}$
UVCluster	rvs161 **ybr108w**	–	–
RNSC$_a$	*sla1 ybr108w*	–	–
RNSC$_s$	*sla1* ypl246c *ygr268c*	2/3	$3.90 \cdot 10^{-04}$
PINCoC	***swe1 hsl7***	2/2	$5.3 \cdot 10^{-04}$
UVCluster	**swe1 hsl7** app1	2/3	$1.91 \cdot 10^{-03}$
RNSC$_{a,s}$	*swe1 hsl7* app1	2/3	$1.91 \cdot 10^{-03}$

generated by *PINCoC*, both UVCLUSTER and RNSC found two groups without any biological meaning. The fourth and fifth clusters are identical for all the methods, except than RNSC$_s$, which was able to score the best p-value for the cluster {*cdc42, cla4, gic2, bni1*} thanks to the protein *bni1*, which does not appear in the corresponding cluster of the other methods. This is the only case in which *PINCoC* does not reach the best p-value score. The seventh cluster generated by *PINCoC* does not contain the protein *app*1, in fact this protein is not involved in the biological process of the other two. Finally, it worth to note that UVCLUSTER and RNSC returned the singleton clusters acf2, yjr083c, ynl086w, ypi236c, ygr268c, ybr108w, and trm5. In our approach this is not possible because of our policy of assigning singleton elements to the most suited clusters. Interestingly, *PINCoC* assigns acf2, yjr083c, and ynl086w to the first cluster, ypi236c, ygr268c, and ybr108w to the sixth cluster, and trm5 to the second one, by obtaining a better p-value. The table points out the very good results of *PINCoC*, comparable with those obtained from the other two methods.

5 Concluding Remarks

We proposed a novel technique to detect significant functional modules in a protein-protein interaction network. The main novelty of the approach is the formalization of the problem of finding dense regions of a PPI network as a co-clustering problem.

The method has two fundamental advantages with respect to other approaches in the literature. The first is that the number of clusters is automatically determined by the algorithm. Furthermore, the problem of ties occurring in protein-protein distances plaguing algorithms based on hierarchical clustering is implicitly solved. As proved by tests carried out on *S. cerevisiae* proteins, the presented method returns partitions that are biologically relevant, correctly clustering proteins which are known to be involved in different biological processes. Future research aims at using *PINCoC* on sets of proteins of other organisms, to characterize proteins whose biological functions are not yet completely known.

References

1. Arnau, V., Mars, S., Marìn, I.: Iterative cluster analysis of protein interaction data. Bioinformatics 21(3), 364–378 (2004)
2. Asur, S., Ucar, D., Parthasarathy, S.: An ensemble framework for clustering protein-protein interaction networks. Bioinformatics 23, i29–i40 (2007)
3. Bader, G., Hogue, H.: An automated method for finding molecular complexes in large protein-protein interaction networks. BMC Bioinformatics 4(2) (2003)
4. Brohèe, S., van Helden, J.: Evaluation of clustering algorithms for protein-protein interaction networks. BMC Bioinformatics 7(488) (2006)
5. Brun, C., Herrmann, C., Guenoche, A.: Clustering proteins from interaction networks for the prediction of cellular functions. BMC Bioinformatics 5(95) (2004)
6. Drees, B.L., Sundin, B., et al.: A protein interaction map for cell polarity development. Journal of Cellular Biology 154, 549–571 (2001)
7. Jain, R.D.A.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
8. King, A.D., Przulj, N., Jurisica, I.: Protein complex prediction via cost-based clustering. Bioinformatics 20(17), 3013–3020 (2004)
9. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. IEEE Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
10. Orlev, N., Shamir, R., Shiloh, Y.: Pivot: Protein interaction visualization tool. Bioinformatics 20(3), 424–425 (2004)
11. Przulj, N., Wigle, D.A., Jurisica, I.: Functional topology in a network of protein interactions. Bioinformatics 20(3), 340–348 (2004)
12. Ucar, D., Asur, S., Çatalyürek, Ü.V., Parthasarathy, S.: Improving functional modularity in protein-protein interactions graphs using hub-induced subgraphs. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS, vol. 4213, pp. 371–382. Springer, Heidelberg (2006)
13. Ucar, D., Parthasarathy, S., Asur, S., Wang, C.: Effective pre-processing strategies for functional clustering of a protein-protein interaction network. In: IEEE Int. Symposium on Bioinformatics and Bioengeneering (BIBE'2005), pp. 129–136 (2005)

Discovering α–Patterns from Gene Expression Data

Domingo S. Rodríguez Baena[1], Norberto Diaz Diaz[1], Jesús S. Aguilar Ruiz[1], and Isabel Nepomuceno Chamorro[2]

[1] Pablo de Olavide University, Seville, Spain
dsrodbae@upo.es, ndiaz@upo.es, direscinf@upo.es
[2] Seville University, Seville,Spain
isabel@lsi.us.es

Abstract. The biclustering techniques have the purpose of finding subsets of genes that show similar activity patterns under a subset of conditions. In this paper we characterize a specific type of pattern, that we have called α–pattern, and present an approach that consists in a new biclustering algorithm specifically designed to find α–patterns, in which the gene expression values evolve across the experimental conditions showing a similar behavior inside a band that ranges from 0 up to a pre–defined threshold called α. The α value guarantees the co–expression among genes. We have tested our method on the *Yeast* dataset and compared the results to the biclustering algorithms of Cheng & Church (2000) and Aguilar & Divina (2005). Results show that the algorithm finds interesting biclusters, grouping genes with similar behaviors and maintaining a very low mean squared residue.

1 Introduction

The DNA microarray technology represents a great opportunity of studying the genomic information as a whole, so we can analyze the relations among thousands of genes simultaneously. The experiments carried out on genes under different conditions produce the expression levels of their transcribed mRNA and this information is stored in DNA chips. The analysis of gene expression data on these biochips is an important tool used in genomic investigations which main objectives range from the study of the functionality of specific genes and their participation in biological process to the reconstruction of diseases conditions and their subsequent prognosis. This huge amount of data has attracted the attention of a lot of researchers because extracting useful information from it represents a big challenge. To deal with it, different statistical and data mining techniques have been applied. Clustering is the most popular approach used in this investigation area [1]. This technique is applied to gene expression data for grouping genes according to their expressions levels under all the experimental conditions, or for creating sets of conditions based on the expression of the genes in study. That is, clustering works only with one dimension: genes or conditions. However, a group of genes can show co–expression under a certain group of conditions, but behave independently under others [2]. Thus, the gene expression data need to be analyzed taking into account the two dimensions at the same time.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 831–839, 2007.

A *bicluster* is a subset of genes that show similar activity patterns under a subset of conditions. The research on biclustering started in 1972 with Hartigan's work [3]. Hartigan's algorithm, named *direct clustering*, divides the data matrix into a certain number of sub–matrices with the minimum variance. In that approach, the perfect bicluster was the sub–matrix formed by constant values, i.e., with variance equal to zero. Another way of searching biclusters is to measure the coherence between their genes and conditions. Cheng & Church [4] introduced a measure, the *mean squared residue* (MSR), that computes the similarity among the expression values within the bicluster. Many researchers have based their works on the ideas of Cheng & Church, trying to improve the method and the results. For instance, Cheng & Church replaced the missing values on datasets by random numbers. Yang et. al [5] [6] solved the problem caused by these random fillings considering only the valid values. As a result of this approach an algorithm named FLOC (Flexible Overlapped biClustering) was designed.

Other alternatives in the searching for biclusters have been studied in the last years. Lazzeroni et. al [7] present the *plaid models*. With these models the data matrix is described as a linear function of layers corresponding to its biclusters. Shamir et al. [8] propose a new method to obtain biclusters based on a combination of graph theoretical and statistical modelling of data. In a recent work [9], a generalization of OPSM (Order–Preserving Submatrix) model, introduced by Ben-Dor et al. [10], is presented. The OPSM model is based on the search of biclusters in which a subset of genes induce a similar linear ordering along a subset of conditions. Some techniques search for structures in data matrix to find biclusters: Gerstein et. al [11] create a method for clustering genes and conditions simultaneously based on the search of "checkerboard" patterns in matrices of gene expression data. In the research of Aguilar et. al [12] an evolutionary technique, based on the search of biclusters following a sequential covering strategy and measuring the mean squared residue, is used.

In this work we propose an algorithm to obtain a specific type of bicluster, that we have called α–pattern, with the maximum number of genes and in which the absolute value of the difference between two expression values of any pair of genes under the same condition is not greater than a threshold, α.

The paper is organized as follows: in Section 2 the new type of pattern is formally characterized and the definitions related to the biclustering method are presented; the algorithm is shown in Section 3; in Section 4, we describe the method used and discuss the experimental results, comparing the quality of those generated by Cheng & Church's and Aguilar & Divina's algorithms; finally, the most interesting conclusions are summarized in Section 5.

2 Definitions

The gene expression data are arranged in matrices. A matrix is defined as a triple $M = (G, C, \ell)$, where G and C are two finite sets referred to as *the set of genes* and *the set of experimental conditions* respectively, and $\ell : G \times C \longrightarrow \Re$ is the *level* function. We will denote the real number $\ell(g, c)$ by $\langle g, c \rangle$, and represents the level of expression of the gene g under the condition c.

Definition 1. *Let $M = (G, C, \ell)$ be a matrix formed by a set of genes, G, and a set of conditions, C. We say that a pair of non-empty sets (I, J) is a α–pattern, if $I \subseteq G$, $J \subseteq C$ and*

$$J = \{c \in C \mid \forall g, g' \in I, \mid \langle g, c\rangle - \langle g', c\rangle \mid \leq \alpha\}$$

The absolute value of the difference between two expression values of any pair of genes in I under the a specific condition from J is not greater than a threshold α.

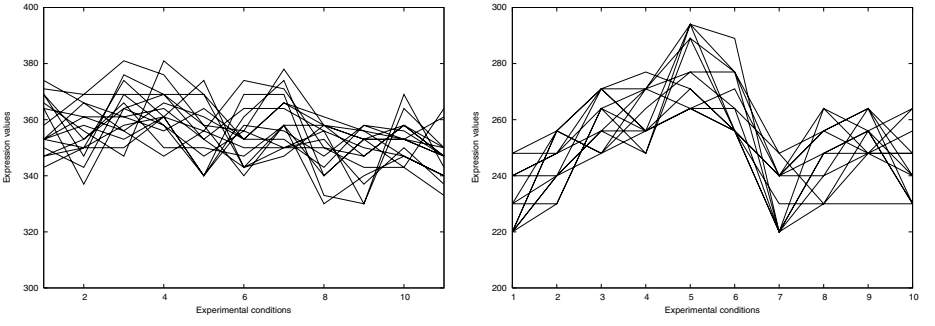


Fig. 1. An example of α–patterns. In both cases we can observe a band, that ranges from 0 up to α, in which the behaviors of the expression values of the genes under a subset of experimental conditions follow similar patterns. The x axis represents the experimental conditions. The y axis represents the gene expression values.

The threshold, α, determines a band in which the expression values of the genes evolve in a similar way across a set of experimental conditions. The co–expression of the genes in these types of bicluster can be observed in Figure 1 (biclusters obtained from Yeast dataset). The expression values in this behavior band can flow along a certain range without many changes, like in the bicluster on the left, or we can find biclusters with more variation in the expression values of the genes, as we can see in the bicluster on the right. In both cases, the shapes of the graphs show the quality of the α–patterns.

To compare the experimental results with those generated by Cheng & Church's algorithm, the *mean squared residue* (MSR) is used. Let (I, J) be a bicluster. The residue R of an element a_{ij} of the bicluster (I, J) is $R(a_{ij}) = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$, where a_{iJ} is the mean of the ith row in the bicluster, a_{Ij} the mean of the jth column and a_{IJ} is the mean of all the elements within the bicluster. The mean squared residue, MSR, of (I, J) is defined as follows:

$$MSR(I, J) = \frac{1}{\mid I \mid\mid J \mid} \sum_{i \in I, j \in J} R^2(a_{ij})$$

This value is indicative of the coherence of values across both rows and columns. The lower the MSR, the stronger the coherence.

1 **Procedure:** αPB
2 **Input:**
3 M (data matrix $M = (G, C, \ell)$)
4 α (maximum difference between two expression values)
5 λ (minimum length of the set of conditions for every bicluster)
6 **Output:**
7 T (final set of biclusters)
8 **Method:**
9 Initialize $T = \emptyset$
11 **For** all bicluster (I, J) with $\mid I \mid = 2$ and $\mid J \mid \geq \lambda$
12 inserted = false
13 **For** every bicluster (I', J') in T
14 **If** $J' \subset J$ and Compatible(I, I', J',α)
15 add I to I'
16 **If** $J \subseteq J'$ and Compatible(I, I', J,α)
17 add I' to I and if $J == J'$ then inserted = true
18 **end For**
19 **If** not inserted
20 add (I, J) to T
21 **end For**
22 **end** αPB

Fig. 2. The α–Pattern Biclustering algorithm

3 Algorithm

Our approach, named αPB (α–Pattern Biclustering), is based on the definition of α–pattern provided above, in Def. 1. The aim is to obtain different biclusters with the maximum number of genes so that all of them have the next two properties:

- the difference between two expression values of any pair of these genes under the same condition in the bicluster is not greater than α.
- the number of conditions of each bicluster is not lower than λ.

The algorithm, illustrated in Figure 2, consists of two parts. In the first part all the valid α–patterns with only two genes are obtained, so that the algorithm analyzes all possible pairs of genes in the data matrix in order to find them, as we can see in line 11. The aim of the second part is to create new biclusters containing more than two genes. The biclusters will be stored in the set T, which is initialized as an empty set in line 9. We have designed a special tree data structure to implement the set T. In this tree, the nodes represent experimental conditions and leaves are sets of groups of genes that have a common group of conditions, i.e., to reach them the same path in the tree have to be followed. This structure is used to minimize the amount of memory used for storing the biclusters (they can be many thousands) and also to reduce the running time.

For each bicluster with two genes, (I,J), obtained in the first part, each bicluster stored in the set T, (I',J'), is analyzed in order to increase their number of genes (line 13). To do this, the groups of conditions and genes of these two biclusters have to be *compatible*.

Table 1. Example of α–patterns with two genes

Bicluster	*Genes*	*Conditions*	*Bicluster*	*Genes*	*Conditions*
$B1$	$\{1,2\}$	$\{2,3,4\}$	$B6$	$\{2,4\}$	$\{1,3,4\}$
$B2$	$\{1,3\}$	$\{1,2,3,4\}$	$B7$	$\{2,5\}$	$\{1,3,4\}$
$B3$	$\{1,4\}$	$\{1,3,4\}$	$B8$	$\{3,4\}$	$\{1,2,3,4\}$
$B4$	$\{1,5\}$	$\{1,3,4\}$	$B9$	$\{3,5\}$	$\{1,2,3,4\}$
$B5$	$\{2,3\}$	$\{2,3,4\}$	$B10$	$\{4,5\}$	$\{1,2,3\}$

Firstly, we check the compatibility between the groups of conditions of both biclusters, i.e., we check if these groups of conditions have one of the following properties (lines 14 and 16): J' $\subset$ J (J' is a subset of J), J $\subset$ J' (J is a subset of J') or J = J' (J and J' are equal).

In the first case, the bicluster (I,J) could provide their genes to the bicluster in set T, (I',J') (line 15). On the contrary , in the last two cases, the set I' could be added to I (line 17). To deal with that increase of the number of genes we have to carry out a second verification. The group of genes of both biclusters, I and I', have to be compatible with the group of experimental conditions, which will be J', if we are in the first case in the preliminary checking, or J, in the last two cases. In the algorithm, the procedure *Compatible(I, I', J ,*α*)* is used to carry out this second verification (lines 14 and 16). Being I and I' groups of genes, J a group of conditions and α the maximum difference between two expressions values, the procedure *Compatible* returns "true" if:

$$\forall g \in I, g' \in I' \text{ and } c \in J, \mid \langle g, c\rangle - \langle g', c\rangle \mid \leq \alpha$$

which means that the absolute value of the difference between two expression values of any pair of genes of I and I', under the same condition of J, is not greater than a threshold α, for all the conditions in J.

Once all the biclusters on T have been analyzed, if the bicluster (I,J) did not add their genes to a bicluster on the tree structure with the same group of conditions, it will be add to the set T (lines 19 and 20).

For a better comprehension of our algorithm we present next a simple example. Consider that the biclusters with only two genes that have been obtained during the first part of the algorithm are those shown in Table 1. The first bicluster is: $B1 = \{\{g_1, g_2\}, \{c_2, c_3, c_4\}\}$. In the first iteration of the algorithm, the tree structure is empty, so $B1$ will be added to T without any modification: $T = \{(c_2, c_3, c_4) \rightarrow (g_1, g_2)\}$.

After that, the tree T has one branch, representing the group of conditions $\{c_2, c_3, c_4\}$ and one leave, representing the group of genes $\{g_1, g_2\}$. Next we are going to process the next bicluster: $B2 = \{\{g_1, g_3\}, \{c_1, c_2, c_3, c_4\}\}$. At this iteration, the tree T has one bicluster stored so we have to determine if the group of conditions and genes of both biclusters are compatible. In the case of the groups of conditions they are compatible because $\{c_2, c_3, c_4\} \subset \{c_1, c_2, c_3, c_4\}$, so in the next step we have to verify if B2 could add its genes to B1. To do this, we have to check if:

$$\forall g \in \{g_1, g_2\}, g' \in \{g_1, g_3\} \text{ and } c \in \{c_2, c_3, c_4\}, \mid \langle g, c\rangle - \langle g', c\rangle \mid \leq \alpha$$

As the two groups of genes have g_1 in common, we don't consider that gene. So we have to verify the previous property with the pair (g_2, g_3). As we can see in Table 1, it exists a bicluster formed by this pair of genes and with the same group of conditions as B1: $B5 = \{\{g_2, g_3\}, \{c_2, c_3, c_4\}\}$, so the property is verified and the genes of B2 can be added to B1 and T will be: $T = \{(c_2, c_3, c_4) \rightarrow (g_1, g_2, g_3)\}$.

Finally, B2 is also added to the tree structure because we didn't add its genes to a bicluster with the same group of conditions in T:

$$T = \{(c_2, c_3, c_4) \rightarrow (g_1, g_2, g_3), (c_1, c_2, c_3, c_4) \rightarrow (g_1, g_3)\}$$

At this point, the tree structure has two biclusters stored.

4 Method and Experimental Results

In this section, the method used to obtain biclusters using the algorithm αPB is described. The aim is to generate biclusters with the maximum number of genes and a low value of MSR. We have developed our experiments with a well known dataset: the *Saccharomyces Cerevisiae* cell cycle expression dataset. The *Yeast* dataset consists of a data matrix composed by 2884 genes (rows) and 17 experimental conditions (columns).

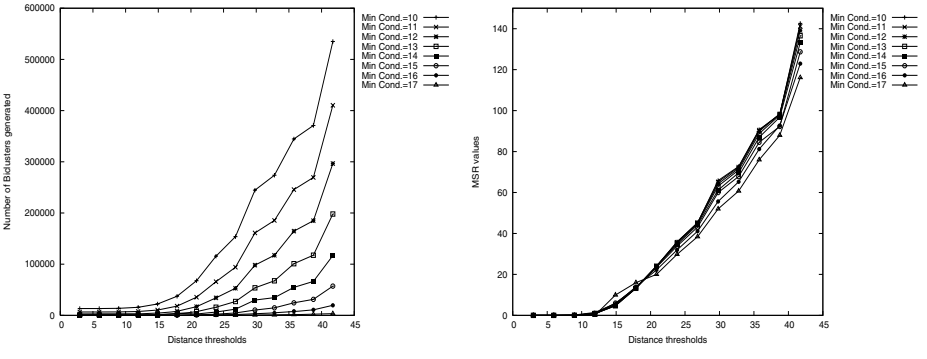


Fig. 3. Number of biclusters (left) and mean of the MSR value of the biclusters generated with different values of α (in X–axis) and λ. It can be observed that the MSR value and the number of biclusters generated increase when the parameters of the αPB algorithm are less restrictive.

The most important parameter of our algorithm is the distance threshold between expression values, α. The parameter α determines the level of similarity between gene behavior across the experimental conditions in the biclusters. To study the influence of the α parameter in the final results, an statistical study has been carried out using the Yeast dataset. We have run a special version of αPB with different values of α and λ parameters to collect information about the number and the mean of the MSR value of final biclusters. The conclusion obtained is that these two magnitudes increase when α and λ are less restrictive. That is, with a high value of the distance threshold and a low number of minimum experimental conditions allowed we can find more biclusters but

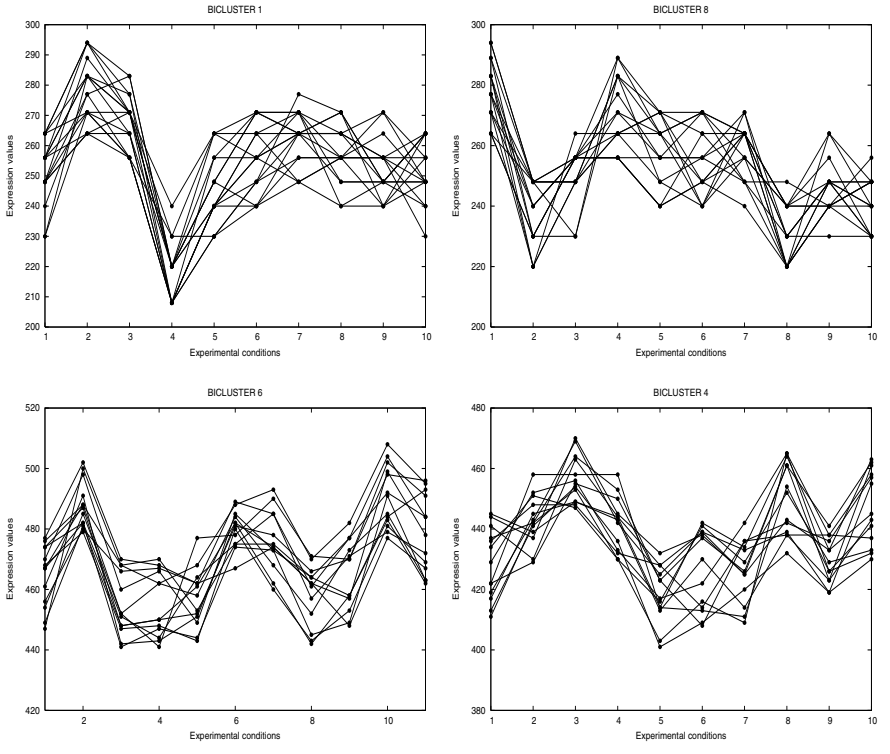


Fig. 4. An example of biclusters generated by αPB. The shapes of the graphs show the quality of biclusters. The y axis range is 100 in all cases.The x axis represents the experimental conditions. The y axis represents the gene expression values.

with less quality. Therefore, we have to achieve a trade–of between these magnitudes. Part of the results of this statistical study is showed in the graphs of Figure 3. In our experiment, we have chosen α=35 and, to make less restrictive the performance conditions, λ=10. With these values, as we can observe in the graphs of Figure 3, we obtain a low mean of MSR, between 80 and 90, and a great number of biclusters: 300000 approximately.

As a result, αPB uses 339421 biclusters with two genes to generates 322150 final biclusters. The maximum number of genes founded in a bicluster is 20, and the mean of the MSR value of all the biclusters is 85.11.

The criterion used to measure the quality of biclusters is the minimum mean squared residue, the maximum number of genes and the maximum number of conditions, in this order. Following these criteria, we have selected the best 100 biclusters among all having a number of genes between 13 and 20. The features of some of them are shown in Table 2. These biclusters present a small MSR value, i.e., it exists a great coherence across both genes and conditions. This similar behavior can be observed in Figure 4. These graphs show the evolution of the expression values of the set of genes under the set of conditions in a behavior band that ranges from 0 up to α. We obtain biclusters

Table 2. Information about biclusters generated by αPB

Bicluster	Genes	Conditions	MSR
1	17	10	73.09
2	15	10	73.48
3	16	11	66.16
4	13	10	72.02
5	20	10	104.95
6	13	11	65.19
7	14	10	71.03
8	16	10	87.41
9	15	10	75.32

Table 3. Performance comparison between CC and αPB

Alg.	MSR	Vol.	Mean Genes	Mean Cond.
CC	204.29	1576,98	166.71	12.09
SEBI	202.68	204.67	13.20	15.44
αPB	81.88	145,27	14.37	10.11

with high number of genes as well, being 20 the maximum value of genes in a bicluster. It is specially interesting that the range of values is very small regarding the range of expression levels (from 0 to 600).

In Table 3, we compare our 100 best biclusters and their average values with those obtained by the algorithms of Cheng & Church (CC) and Aguilar & Divina (SEBI). As we can observe, αPB obtains better results with regard to the MSR value. The averaged volume, i.e., the number of genes multiplies by the number of conditions, and the average of conditions in biclusters are lower. The average of genes is lower than the same measure obtained by CC but greater if it is compared with SEBI. The most interesting property of α–patterns found by αPB is that it provides biclusters with very low mean squared residue in comparison to the CC and SEBI algorithms, while maintaining a good number of genes, between 13 and 20.

5 Conclusions

In this work we present a new technique to discover a certain type of biclusters in gene expression data. These biclusters, named α–patterns, are based on the distance between the expression values of genes. The distance threshold α determines a band in which the expression values of a subset of genes have similar behavior under a subset of conditions. Our approach, named αPB, provides a group of different biclusters with highly–related genes and very low mean squared residue. Results show interesting biclusters in comparison to Cheng & Church approach (CC, based on a greedy strategy) and Aguilar & Divina's ones (SEBI, based on an evolutionary technique) . Our approach obtains biclusters with much less number of genes than CC and the lowest mean squared residue.

References

1. Dougherty, E., Barrera, J.: Inference from Clustering with Application to Gene-Expression Microarrays. Journal of Computational Biology 9(1), 105–126 (2002)
2. Madeira, S., Oliveira, A.: Biclustering Algorithms for Biological Data Analysis: A Survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
3. Hartigan, J.A.: Direct clustering of a data matrix. Journal of the American Statistical Association 67(337), 123–129 (1972)
4. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology, pp. 93–103 (2000)
5. Yang, J., Wang, W.: Improving Performance of Bicluster Discovery in a Large Data Set. In: 6th ACM International Conference on Research in Computational Molecular Biology, Poster (2002)
6. Yang, J., Wang, W.: Enhanced biclustering on expression data. In: 3rd IEEE Conference on Bioinformatics and Bioengineering, pp. 321–327 (2003)
7. Lazzeroni, L.: Owen: Plaid models for gene expression data. Technical Report Stanford University (2000)
8. Shamir, R., Sharan, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics 19(Suppl. 1), 136–144 (2002)
9. Liu, J., Yang, J.: Biclustering in Gene Expression Data by Tendency. In: IEEE Computational Systems Bioinformatics Conference, pp. 183–193 (2004)
10. Ben-Dor, A., Chor, B.: Discovering local structure in gene expression data: The Order Preserving Submatrix Problem. In: 6th ACM International Conference on Research in Computational Molecular Biology (2002)
11. Gerstein, M., Chang, J.: Spectral Biclustering of Microarray Data: Coclustering Genes and Conditions. Journal Genome Research 13(4), 703–716 (2003)
12. Aguilar, J.S., Divina, F.: Evolutionary Biclustering of Microarray Data. In: 3rd European Workshop on Evolutionary Bioinformatics (2005)

Biclusters Evaluation Based on Shifting and Scaling Patterns

Juan A. Nepomuceno[1], Alicia Troncoso Lora[2], Jesús S. Aguilar–Ruiz[2], and Jorge García–Gutiérrez[1]

[1] Department of Computer Science,
University of Sevilla, Sevilla, Spain
{janepo, jgarcia}@lsi.us.es
[2] Area of Computer Science, School of Engineering,
Pablo de Olavide University, Sevilla, Spain
{ali, aguilar}@upo.es

Abstract. Microarray techniques have motivated the develop of different methods to extract useful information from a biological point of view. Biclustering algorithms obtain a set of genes with the same behaviour over a group of experimental conditions from gene expression data. In order to evaluate the quality of a bicluster, it is useful to identify specific tendencies represented by patterns on data. These patterns describe the behaviour of a bicluster obtained previously by an adequate biclustering technique from gene expression data. In this paper a new measure for evaluating biclusters is proposed. This measure captures a special kind of patterns with scaling trends which represents quality patterns. They are not contemplated with the previous evaluating measure accepted in the literature. This work is a first step to investigate methods that search biclusters based on the concept of shift and scale invariance. Experimental results based on the yeast cell cycle and the human B-cell lymphoma datasets are reported. Finally, the performance of the proposed technique is compared with an optimization method based on the Nelder-Mead Simplex search algorithm.

Keywords: Gene expression data, biclustering, shifting and scaling patterns, unconstrained optimization.

1 Introduction

In the last few years microarrays techniques have generated a great amount of biological information. Microarray data can be represented by a numerical matrix with its columns corresponding to experimental conditions and rows associated with genes. Thus the element (i, j) is the expression level of the gene i under the specific condition j. Data mining techniques have been successfully applied to gene expression data in order to discover subtypes of diseases, identification of functional grouping of genes, etc. Clustering techniques have been applied to microarray data [1] in order to identify groups of genes that show similar expression patterns. Most of clustering models have been focused on discovering

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 840–849, 2007.

clusters embedded in a subset of dimensions, because relevant genes are not necessary related to every condition [2]. This problem is known as *biclustering or subspace clustering*. Thus, the goal of biclustering techniques is to extract subgroups of genes with similar behavior under specific subgroups of conditions [3]. This is a vital task from a biomedical point of view, since it is the first step in order to discover networks of genes interaction.

Biclustering problem is a NP-hard problem [4], therefore different techniques use heuristics approaches in order to find biclusters, for example evolutionary algorithms [5,6,7]. These methods are based on a measure to evaluate the quality of biclusters, with the *Mean Squared Residue* (MSR) [8] the most important measure for assessing the quality of biclusters. For this reason, bicluster evaluation is a vital task for searching patterns in biological data. MSR evaluation measure is based on computing the arithmetic means of the values in each row, column, and the full matrix, and the numerical differences among the data. However, it have been proved that MSR is effective for recognizing biclusters with shifting patterns but not some patterns with scaling trends, in spite of representing quality patterns [9]. A bicluster has a shifting pattern when its values vary in the addition of a constant value, and scaling pattern when its values vary in the multiplication of a constant value. A *perfect bicluster* is considered as the one which follows exactly a perfect shifting and scaling pattern [4]. Consequently, it is interesting to study the behavior or tendencies in a bicluster in order to establish a new quality measure through the degree of similarity with its corresponding perfect bicluster.

This fact represents the main motivation of this work where a new measure for evaluating biclusters is proposed. We apply a classical optimization method to solve a least squared statistical estimation problem in order to build the perfect bicluster of a bicluster. After that, the value of the optimization function in the convergence point is the value for the measure of it. If a bicluster presents perfect shifting and scaling patterns, it will be a perfect bicluster itself and its measure will be zero. Although the main task about biclustering problem is to find good biclusters from a microarray, this work is relevant in order to investigate methods that search biclusters based on the concept of shift and scale invariance. First, the problem is formulated from a mathematical point of view leading to an unconstrained nonlinear optimization problem. Later, the problem is solved by a classical Quasi-Newton method. Finally, experimental results obtained from biclusters on the yeast cell cycle and the human B-cell lymphoma datasets are reported. The performance of the proposed method is compared with a search technique based on the Nelder-Mead Simplex algorithm.

The paper is organized as follows: Section 2 presents basic concepts on patterns from gene expression data. A brief overview on unconstrained optimization techniques is shown in Section 3. The formulation of the problem is described in Section 4. Section 5 reports some results obtained from the application of two techniques to two real datasets. Finally, the main conclusions of the paper are outlined.

2 Shifting and Scaling Patterns

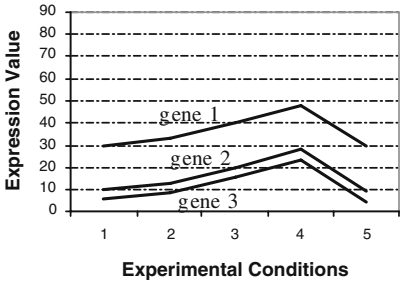
Given a bicluster, the shifting and scaling patterns can be formally defined. A microarray M is a real matrix composed by N genes and M conditions. The element (i, j) of the matrix is represented by $v_{i,j}$. A bicluster B is a submatrix of M composed by $n \leq N$ rows and $m \leq M$ columns. The element (i, j) of the bicluster B is represented by $w_{i,j}$.

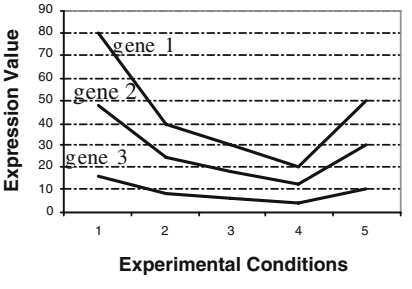
A group of genes has a shifting pattern when the values $w_{i,j}$ vary in the addition of a value β_i. Analogously, a bicluster has a scaling pattern when the values $w_{i,j}$ vary in the multiplication of a value α_i. The values β_i and α_i are fixed for all the genes. Formally, a bicluster shows a shifting or scaling pattern respectively when it follows the expressions (1) or (2) respectively:

$$w_{i,j} = \pi_j + \beta_i \tag{1}$$

$$w_{i,j} = \pi_j \times \alpha_i \tag{2}$$

where π_j is a typical value for the gene j and fixed for all the conditions.





$$\begin{bmatrix} 30 & 10 & 5 \\ 33 & 13 & 8 \\ 40 & 20 & 15 \\ 48 & 28 & 23 \\ 29 & 9 & 4 \end{bmatrix} = \begin{bmatrix} 30+0 & 10+0 & 5+0 \\ 30+3 & 10+3 & 5+3 \\ 30+10 & 10+10 & 5+10 \\ 30+18 & 10+18 & 5+18 \\ 30-1 & 10-1 & 5-1 \end{bmatrix} \qquad \begin{bmatrix} 80 & 48 & 16 \\ 40 & 24 & 8 \\ 30 & 18 & 6 \\ 20 & 12 & 4 \\ 50 & 30 & 10 \end{bmatrix} = \begin{bmatrix} 10\times 8 & 6\times 8 & 2\times 8 \\ 10\times 4 & 6\times 4 & 2\times 4 \\ 10\times 3 & 6\times 3 & 2\times 3 \\ 10\times 2 & 6\times 2 & 2\times 2 \\ 10\times 5 & 6\times 5 & 2\times 5 \end{bmatrix}$$

Fig. 1. Bicluster with a) shifting patterns, b) scaling patterns

Figure 1a) presents a bicluster that contains a shifting pattern. Shifting patterns represent related genes that show the same shape and slope. It can be observed that the genes start with different initial values. Thus, shapes of the graphs are similar, but values are not equal. Figure 1b) presents a bicluster that contains a scaling pattern. In this case, scaling patterns represent related genes showing the same shape, but different slopes. It can be noted that changes more abrupt for one gene than for the other are shown.

In a general case, an element of a bicluster showing both types of patterns can be defined as:

$$w_{i,j} = \alpha_i \times \pi_j + \beta_i + \varepsilon_{i,j} \tag{3}$$

where $\varepsilon_{i,j}$ is the error that the patterns have for the value $w_{i,j}$ of the bicluster considered.

A bicluster is a perfect bicluster when the value of $\varepsilon_{i,j}$ is equal to zero for all values $w_{i,j}$ of the bicluster.

3 Unconstrained Optimization Techniques

Unconstrained optimization techniques are used to search local minima in optimization problems whose objective function is not subject to equality and inequality constraints.

An unconstrained optimization problem can be defined as:

$$min\ f(x)$$

where $x \in \mathbb{R}^n$ is a vector of real variables and $f : \mathbb{R}^n \mapsto \mathbb{R}$ is a linear or nonlinear scalar function.

There is a great number of methods to solve unconstrained optimization problems. Nowadays, unconstrained optimization problems can be classified in two groups: search methods and gradient methods. Search methods [10] use only function evaluations and these approaches are most suitable for problems that are very nonlinear or have a great number of discontinuities. Simplex search methods are based on searching the local minima inside a particular region or simplex. A simplex in n-dimension space is characterized by $n + 1$ distinct vectors that are its vertices. At each iteration, the objective function is evaluated in a new point generated inside the simplex, which is compared with value of the function at vertices of the simplex. One of the vertices could be replaced by the new point. The process is repeated until the diameter of the simplex is less than a specified tolerance.

Gradient methods [11] are generally more efficient when the first derivative of the objective function is continuous. The search direction to locate the minimum is proportional to the gradient of the objective function as follows [1]:

$$x_{k+1} = x_k - \alpha_k \cdot \nabla f(x_k) \tag{4}$$

where α_k is the step–length parameter and x_k is the variable x at iteration k.

The parameter α_k is obtained by a line–search method. The line–search approach consists in solving a minimization problem in one dimension. This problem can be formulated as follows:

$$min\ \phi(\alpha)$$

where $\alpha \in \mathbb{R}$ and $\phi(\alpha) = f(x_k + \alpha \cdot \nabla f(x_k))$ with x_k and $\nabla f(x_k)$ fixed.

[1] $\nabla f(x_k)$ is the value of gradient of function f in x_k point.
$\nabla f(x_k) = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$, $\frac{\partial f}{\partial x_i}$ represents the derivative of f respect x_i variable.

Newton methods are higher order gradient methods. This is due to the use of second order information. These methods are only really suitable when the second order information is readily and easily calculated, because calculation of such is computationally expensive. In this case, the search direction can be written as follows:

$$x_{k+1} = x_k - \alpha_k \cdot H_k^{-1} \cdot \nabla f(x_k) \tag{5}$$

where H_k^{-1} is the inverse of the Hessian matrix at point x_k.

Quasi-Newton methods are Newton methods, which use an approximation to the inverse of the matrix H_k as an alternative to calculate it directly. Different Quasi-Newton methods are based on different approximations of the inverse of the matrix H_k [12,13].

4 Formulation of the Problem

The objective of the problem is to determine the shifting and scaling patterns of a certain bicluster. It is supposed that biclusters are obtained previously by appropriate biclustering techniques.

The objective function is defined by the mean squared error (MSE) as follows:

$$MSE = \frac{1}{n \cdot m} \sum_{i=1}^{n} \sum_{j=1}^{m} \varepsilon_{i,j}^2 \tag{6}$$

where $\varepsilon_{i,j}$ is defined in Eq. 3. It can be noted that the error $\varepsilon_{i,j}$ depends on the shifting patterns β_i, scaling patterns α_i and the typical value for each gene π_j.

Thus, the problem can be formulated as the following unconstrained optimization problem:

$$min \quad f(\overrightarrow{\alpha}, \overrightarrow{\beta}, \overrightarrow{\pi}) \tag{7}$$

where $\overrightarrow{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$, $\overrightarrow{\beta} = (\beta_1, \ldots, \beta_n) \in \mathbb{R}^n$, $\overrightarrow{\pi} = (\pi_1, \ldots, \pi_m) \in \mathbb{R}^m$ and $f : \mathbb{R}^{2n+m} \longmapsto \mathbb{R}$ is defined by the MSE (Eq. 6).

The result of this optimization problem is the optimal point and the value of the function on it. Shifting and scaling patterns for the bicluster are built with $\overrightarrow{\alpha}, \overrightarrow{\beta}$ and $\overrightarrow{\pi}$ values. The quality of the bicluster is established with the value of the objective function on the solution.

This unconstrained optimization problem has been solved by using a Quasi–Newton method. The Hessian matrix has been approximated by using the formula of Shanno [13] and the step-length parameter has been determined by a line–search technique. Also, this optimization problem has been solved by using the Nelder-Mead Simplex search algorithm in order to establish a comparison.

5 Experiments

An unconstrained optimization technique based on the Quasi-Newton method has been applied to solve the proposal problem. Shifting and scaling patterns,

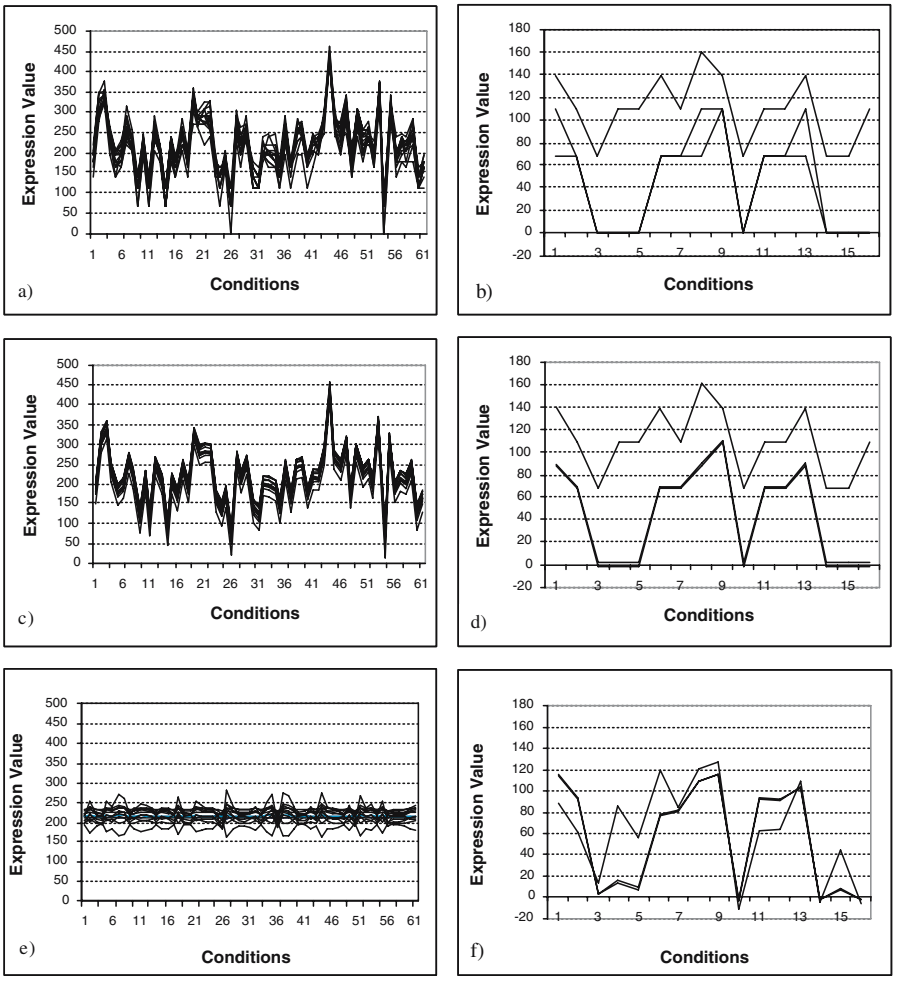


Fig. 2. a), b) Two yeast bicluster leading to the worst and best patterns respectively, c), d) patterns obtained with Quasi-Newton method and e), f) patterns obtained with Nelder-Mead Simplex algorithm

that is to say, the perfect bicluster which approximates the original bicluster, and the value of objective function like a quality measure are obtained for each bicluster.

Original biclusters have been obtained from the work recently published in [5]. These biclusters have been built from a biclustering technique based on an evolutionary algorithm applied to two well-known datasets: yeast *Saccharomyces cerevisiae* cell cycle expression dataset [14]; and the human B-cells expression data [15]. The Yeast dataset contains 2884 genes and 17 experimental conditions and the Human dataset consists of 4026 genes and 96 conditions. The proposed

technique has been applied over the one hundred biclusters obtained in [5]. Results obtained for the biclusters leading to the worst and the best shifting and scaling patterns in both datasets are reported.

Figures 2a) and 2b) present two biclusters from Yeast dataset leading to the worst and the best shifting and scaling patterns obtained by the Quasi-Newton optimization method (Figures 2c) and 2d)), in the sense of the highest and the lowest value for the evaluation function. These two biclusters are composed by 14 and 3 genes and 61 and 16 conditions, respectively. Notice that the expression values for several genes over certain conditions are the same (black thick lines).

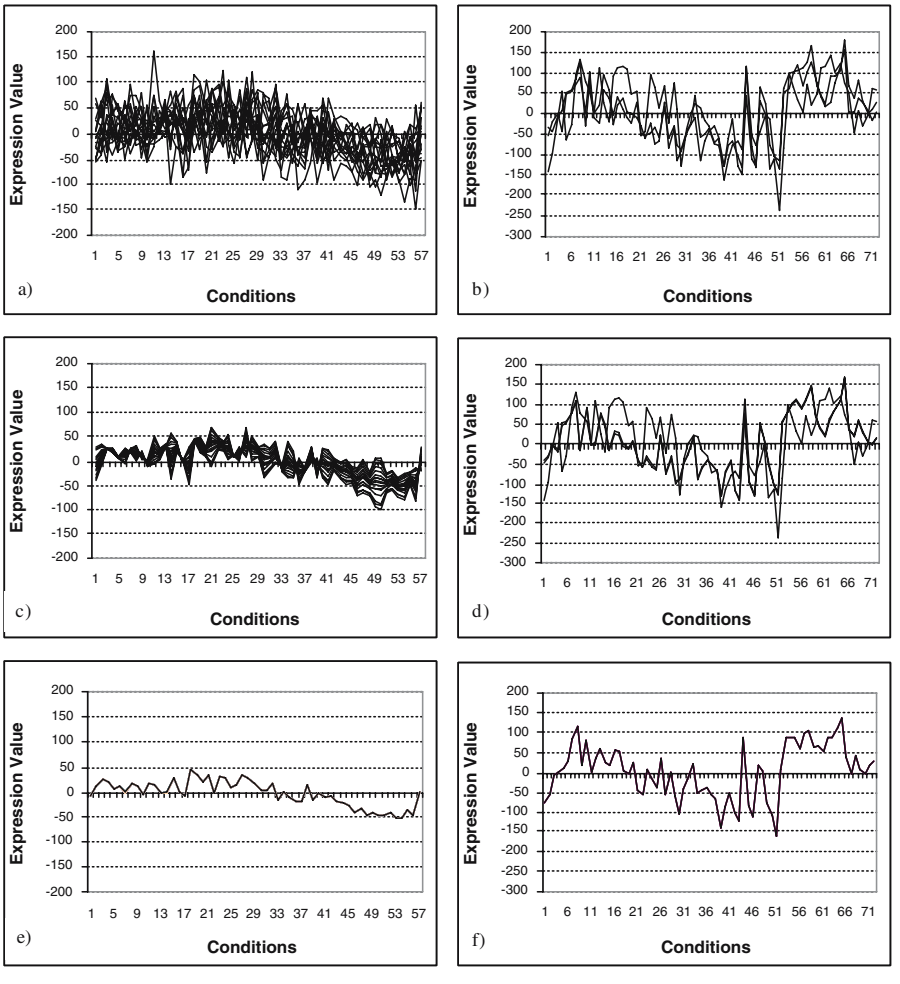


Fig. 3. a), b)Two human bicluster, leading to the worst and best patterns respectively, c), d) patterns obtained with Quasi-Newton method and e), f) patterns obtained with Nelder-Mead Simplex algorithm

Table 1. Comparison between two optimization methods used to build shifting and scaling patterns in biclusters: Quasi–Newton method and Nelder-Mead Simplex algorithm

	number of iterations		**time in seconds**		**obj. function**	
	Yeast	Human	Yeast	Human	Yeast	Human
Quasi-Newton method	67,59	193,67	5,08	83,73	13,31	30,52
N-M Simplex algorithm	49264,20	50000	69,15	226,06	66,431	35,05

Its corresponding shifting and scaling patterns are shown in Figures 2c) and 2d). The final value of the error function defined by the MSE (Eq. 6) is equal to 14.51 for bicluster on the left and 7.16 for bicluster on the right, respectively. A good quality of the discovered patterns can be observed in both biclusters. Figures 2e) and 2f) show the patterns obtained by the optimization method based on the Nelder-Mead Simplex algorithm for two biclusters. Notice that these shifting and scaling patterns obtained are worst than the first ones, in the sense that they adjust the shape of the original bicluster in a worst way.

Figures 3a) and 3b) present two biclusters from Human dataset leading to the worst and the best shifting and scaling patterns, left and right respectively. These biclusters are constituted for 17 genes and 57 conditions and 3 genes and 72 conditions, respectively. A bad quality of the built patterns can be observed in the Figure 3d), in spite of the bicluster leads to the best patterns. It is due to the low number of genes and the irregular behaviour of this bicluster. Obviously, biclusters with an uniform behaviour provide better shifting and scaling patterns than those with no inherent tendency. Figures 3e) and 3f) present the patterns obtained from the application of the Nelder-Mead Simplex method for two biclusters. These patterns have 17 and 3 genes, as original bicluster, but expression values for genes over all the conditions are the same, as only a black thick line can be distinguished. Obviously, these shifting and scaling patterns are worst than the previous ones obtained with the proposed method based on the Quasi-Newton method.

Finally, in Table 1 a comparison is made between the two used techniques, Quasi–Newton algorithm and Nelder-Mead Simplex search method. Table 1 shows the most representative parameters of optimization process for two datasets. It can be observed the average of the iterations number, the CPU time and the value of the objective function on one hundred biclusters obtained from yeast cell cycle microarray and one hundred biclusters from human B-cells microarray. Notice the highest cost in time (in seconds) and in number of iterations and the worst patterns are obtained by Nelder-Mead Simplex search approach.

6 Conclusions

An unconstrained optimization technique has been applied in order to build shifting and scaling patterns from biclusters. The method has been tested over biclusters obtained from two different real datasets: yeast cell cycle and human

B-cells. Results have shown that the proposed approach has a good performance for finding shifting and scaling patterns of a given bicluster. The proposed technique has been compared with an optimization method based on the Nelder-Mead Simplex algorithm showing better results with regarding to the patterns found and the CPU time.

Future works will be focused on the comparison between different biclustering algorithms using the proposal measure in order to establish which one is the best. Some more actual microarrays taken from PNAS journal will be used. Biclusters obtained by others biclustering algorithms such as Cheng-Church, ISA, OPSM, etc, will be also used for this proposal. On the other hand, we will also study the possibility of new biclustering tecniques based on the concept of shift and scale invariance.

Acknowledgments. Thanks are due to the Spanish CICYT (TIC2004-00159) and Junta de Andalucía (P05-TIC-00531) for sponsoring this research.

References

1. Ben-Dor, A., Shamir, R., Yakhini, Z.: Clustering Gene Expression Patterns. Journal of Computational Biology 6, 281–297 (1999)
2. Wang, H., Wang, W., Yang, J., Yu, P.S.: Clustering by Pattern Similarity in Large Data Sets. In: ACM SIGMOD International Conference on Management of Data, pp. 394–405 (2002)
3. Tanay, A., Sharan, R., Shamir, R.: Discovering Statistically Significant Biclusters in Gene Expression Data. Bioinformatics 18, 196–205 (2002)
4. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1, 24–45 (2004)
5. Divina, F., Aguilar-Ruiz, J.S.: Biclustering of Expression Data with Evolutionary Computation. IEEE Transactions on Knowledge & Data Engineering 18(5), 590–602 (2006)
6. Bryan, K., Cunningham, P., Bolshakova, N.: Biclustering of Expression Data Using Simulated Annealing. In: IEEE Symposium on Computer-Based Medical Systems, pp. 383–388 (2005)
7. Mitra, S., Banka, H.: Multi-objective evolutionary biclustering of gene expression data. Pattern Recognition 39(12), 2464–2477 (2006)
8. Cheng, Y., Church, G.M.: Biclustering of Expression Data. In: Proceedings of the 8th International Conference on Itelligent. Systems for Molecular Biology, La Jolla, CA, pp. 93–103 (2000)
9. Aguilar-Ruiz, J.S.: Shifting and Scaling Patterns from Gene Expression Data. Bioinformatics 21(20), 3840–3845 (2005)
10. Nelder, J.A., Mead, R.: A Simplex Method for Function Minimization. Computer J. 7, 308–313 (1965)
11. Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear Programing: Theory and Algorithms. John Wiley and sons, Chichester (1993)
12. Fletcher, R.: A New Approach to Variable Metric Algorithms. Computer Journal 13, 317–322 (1970)

13. Shanno, D.F.: Conditioning of Quasi-Newton Methods for Function Minimization. Mathematics of Computing 24, 647–656 (1970)
14. Cho, R., et al.: A Genome-Wide Transcriptional Analysis of the Mitotic Cell Cycle. Molecular Cell 2, 65–73 (1998)
15. Alizadeh, A.A., et al.: Distinct Types of Diffuse Large b-cell Lymphoma Identified by Gene Expression Profiling. Nature 403, 503–511 (2000)

A Deterministic Model to Infer Gene Networks from Microarray Data

Isabel Nepomuceno-Chamorro, Jesús S. Aguilar–Ruiz, Norberto Díaz–Díaz, Domingo S. Rodríguez–Baena, and Jorge García

BIGS BioInformatics Group Seville
University of Seville
Pablo de Olavide University
{isabel,ndiazjgarcia}@lsi.us.es, jsagurui,dsrodbae@upo.es

Abstract. Microarray experiments help researches to construct the structure of gene regulatory networks, i.e., networks representing relationships among different genes. Filter and knowledge extraction processes are necessary in order to handle the huge amount of data produced by microarray technologies. We propose regression trees techniques as a method to identify gene networks. Regression trees are a very useful technique to estimate the numerical values for the target outputs. They are very often more precise than linear regression models because they can adjust different linear regressions to separate areas of the search space. In our approach, we generate a single regression tree for each genes from a set of genes, taking as input the remaining genes, to finally build a graph from all the relationships among output and input genes. In this paper, we will simplify the approach by setting an only seed, the gene ARN1, and building the graph around it. The final model might gives some clues to understand the dynamics, the regulation or the topology of the gene network from one (or several) seeds, since it gathers relevant genes with accurate connections. The performance of our approach is experimentally tested on the yeast Saccharomyces cerevisiae dataset (Rosetta compendium).

1 Introduction

In a microarray experiment the mRNA expression level of several thousands of genes is measured systematically, and this allows researchers to analyze data by using statistics or data mining techniques. Lately, a new approach for the interpretation of high–throughput gene expression has been proposed, *gene networks* which is based on the study of network topologies and it addresses a variety of biological systems, metabolic networks, networks of proteins, protein interactions, etc. A gene network is a direct graph, in which each node represents a gene and the relationship among different genes is represented by the edges. Several methods have been proposed to model such gene network that can be derived from gene expression data. These methods identify genes with similar expression in different situations and cluster them according to this similarity

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 850–859, 2007.

[1]. One of the differences between such models is whether they are deterministic or probabilistic. Amongst deterministic models, we consider: discrete boolean network [2], differential equations [3] as continuous method and petri nets [4] as hybrid method. The first is more computationally tractable and less accurate than continuous methods. The hybrid method can model systems with continuous flows and the discrete part models the logic functioning. Amongst probabilistic models, we consider: probabilistic boolean networks [5]; co-expression graphs [6], which are easy to interpret but they cannot distinguish direct from indirect dependencies between genes; conditional independence models as sparse Gaussian Graphic Models [7] and mutual information [8], these methods cannot estimate relationships if the number of variables is large compared to the number of samples; and finally bayesian networks [9].

In regression domains, Quinlan presented the system M5 [10]. It builds multivariate trees using linear models at the leaves. In the pruning phase for each leaf a linear model is built. Recently Witten and Frank have presented M5' in [11], a rational reconstruction of Quinlan's M5 algorithm. M5' first constructs a regression tree by recursively splitting the instance space using tests on single attributes that maximally reduce variance in the target variable. After the tree has been grown, a linear multiple regression model is built for every inner node, using the data associated with that node and all the attributes that participate in tests in the subtree rooted at that node. Then the linear regression models are simplified by dropping attributes, if this results in a lower expected error on future data (more specifically, if the decrease in the number of parameters outweighs the increase in the observed training error). After this has been done, every subtree is considered for pruning. Pruning occurs if the estimated error for the linear model at the root of a subtree is smaller or equal to the expected error for the subtree. After pruning terminates, M5' applies a smoothing process that combines the model at a leaf with the models on the path to the root to form the final model that is placed at the leaf. Also Karalic studied the influence of using linear regression in the leaves of a regression tree [12]. As in the work of Quinlan, Karalic shows that it leads to smaller models with increase of performance. Torgo has presented an experimental study about functional models for regression tree leaves [13]. Later, the same author [14] presented the system Regression Tree (RT). RT is a system which is able to use several functional models at the leaves, including partial linear models. RT builds and prunes a regular univariate tree. Then at each leaf a linear model is built using the examples that fall at this leaf. In the regression setting few works consider multi–variable splits. One of them has been presented by Li et al. in [15], where decision nodes contain linear regressions and data is split according to the sign of the residuals.

Using regression trees has the goal of showing that functional relationships can be found in the genetic network obtained from those trees. We propose to take each gene as output for the regression tree and the remaining genes as input. The output gene and its input genes define a gene–gene interaction, i.e., an edge in the graph. Therefore, the topology of the gene network is based on the accuracy of linear regressions.

The paper is organized as follows: Section 2 presents other approaches that use regression techniques to infer gene networks; Section 3 presents the approach based on regression trees; the experiments are discussed in Section 4; finally, the most interesting conclusions are summarized in Section 5.

2 Related Work

There is a wide range of approaches available to build gene network up. Several of this methods are mentioned before. The novelty of this work is using regression technique to infer gene–gene relationship. As mentioned above, our approach is based on the accuracy of linear regressions and we use parallel regressions of each gene onto the other genes by trees. This involves splitting the instances space and we use regression trees. Therefore our approach can be classified as a deterministic model.

The most significant approaches using bayesian network are [16,17]. The latter involves learning a probabilistic model from partially observed data by using SEM (Stochastic Expectation and Maximization) algorithm. First, the algorithm is initialized by using a standard expression clustering technique to choose assignments of genes to process. Second, successive iterations of the algorithm are run to learn the graph topology and the parameters of each conditional probability distribution. Learning graph topologies is much harder than learning parameters and [16] uses regression trees to learn those graph topologies, i.e. they use this techniques for each biological process (represented by a subset of genes from the microarray) that maximizes bayesian score. However, unlike Battle, we use the regression trees to study the relationship between genes and learning gene–gene interactions from the whole microarray data.

3 Approach

The goal of our approach is to discover distinct expression patterns in which a set of genes is associated to each gene. The groups of ORFs are obtained from their prediction ability by means of regressions. We use regression trees because these representations work like several linear regressions at the same time, each of them identified by a leaf of the tree. The main advantage of this methodology is that each regression is specialized in a specific area of the search space, and hence the regression tree is generally more accurate than a global linear regression.

3.1 Building Regression Trees

A decision tree induction algorithm builds an initial tree without pruning. We use in this algorithm smoothed predictions to compensate sharp discontinuities that will inevitably appear between adjacent linear models. This is an usual problem for models constructed from a small number of training instances.

We use as splitting criterion of the search space the attribute of the data set that maximizes the expected error reduction. This error is called SDR for Standard Deviation Reduction and is calculated by:

$$SDR = sd(T) - \sum_i \frac{|T_i|}{|T|} \times sd(T_i)$$

where T_1, T_2,...,T_k are the sets that result from splitting the node according to the chosen attribute, and *sd* is the standard deviation.

The BMT (Build a Model Tree) and SP (SPlit node in branchs) algorithms (see Algorithms 1 and 2, respectively) present the pseudo–code for the regression tree algorithm explained above. Inputs of the BMT algorithm are a gene and a matrix that represents the microarray experiments. The output is a regression tree that represents the input.

3.2 Building the Gene Network

Next, we build a gene network from the regression trees, i.e., from a representation of each gene expression. The idea behind our approach is to build a regression tree for each input gene from the dataset. Every regression tree has internal genes, which divide the search space, and regression genes, which are involved in the linear regressions. They are considered as output genes of the algorithm SP. Internal genes (IG) split the search space according to their values and regression genes (RG) adjust the linear regressions in the subspace defined by internal genes. When the algorithm is run, a direct labelled graph defined from dependencies among output and input genes are provided by our approach.

A graph is defined as a tuple *(G,E)* of nodes *G* and edges *E*. An edge is an ordered pair of nodes *(gene1, gene2)* and we can interpret that the first gene or node encodes a knows transcription factor that binds to the promoter of the second gene. We build a forest of trees and we learn the gene network in terms of input–output dependency of genes between input genes and output genes (i.e. IG and RG) of each regression tree. So the relationships among output and input genes define the edges of the graph that represents the topology of the gene networks.

The algorithm BGN (see Algorithm 3) builds the forest of regression trees and the gene network step by step. The construction of gene networks from expression data is a difficult and problematic task due to the huge amount of

Algorithm 1. BMT - Build a model tree

INPUT *matrix*: a set of instances in a data set (gene expression data)
OUTPUT *RT*: Regression Tree
begin
  SD ← Calculate the standard deviation of the class values of *matrix*
  RT ← SP(*matrix*,SD)
end

Algorithm 2. SP - Split node in branchs

INPUT *matrix, SD*: gene expression data and standard deviation of the class values of a set of instances
OUTPUT *RT*: Regression Tree
begin
  sd ← standard deviation of the class values of *matrix*
  if sd $< 0.05 \times SD$ **then**
    type of node ← LEAF
  else if sd $> 0.05 \times SD$ **then**
    type of node ← INTERMEDIATE
    for all genes **do**
      for all possible split positions of the gene **do**
        Calculate the attribute's SDR
      end for
    end for
    gene ← attribute with maximum SDR
    $matrix_l$ ← conditions from the matrix to the left of the gene's split
    $matrix_r$ ← conditions from the matrix to the right of the gene's split
    gene.left ← SP($matrix_l$, SD)
    gene.right ← SP($matrix_r$, SD)
  end if
end

data, thousand of genes [18] and [19]. This implies deciding which genes must be included in the learning process. As an example, Peér et al. work with 565 initial genes and excludes the remaining 5751 in the yeast study.

Peña et al. [20] construct a gene network from seed genes. In this paper, we adopt the same idea, because of its simplicity, although we have increased the gene network by means of an iterative process. The algorithm BGN starts with a set S of genes, which at the beginning has only one seed. For this seed–gene, we build the regression tree and the output genes (from the regression tree) are added to the set S of genes. The genes recently added are then used as output to enlarge the levels of the graph. This is an iterative process, in which the number of iterations is calculated by experimentation.

To measure how frequent is used a local regression by experimental conditions, the *support* of a leaf is taken into account: $Support = \frac{LM*100}{N}$ where N is the number of instances in the data set and LM is the number of conditions that are classified in the specific area of the search space covered by the regression. The estimate error is defined as follows:

$$error = \frac{100 \times MSE}{global\ absolute\ deviation}$$

where MSE is the mean squared error defined by:

$$MSE = \frac{(p_1 - a_1)^2 + ... + (p_n - a_n)^2}{n}$$

Algorithm 3. BGN - Build Gene Network

INPUT *matrix*: gene expression data
OUTPUT GR, GN: Gene rank and Gene Network
begin
  GR ← ∅
  GN ← ∅
  SeedGenes ← {*ARN*1}
  for all attribute–gene from *SeedGenes* **do**
    RT ← BMT(matrix, attribute–gene) //build a regression tree
    GR ← update the rank of genes GR from RT
    GN ← update the gene network GN from RT
    SeedGenes ← ∅
    SeedGenes ← update the SeedGenes from RT
  end for
end

with p_1, p_2,...,p_n being the predicted values on the test instances, a_1, a_2,...,a_n being the real values and n is the number of attributes.

Among all the trees in the forest, we consider those whose relative error is lower than a threshold value. In a regression tree generated for an output gene, we only consider the branches of the tree whose support is greater than a threshold value, i.e., linear regressions with small number of conditions met are not accepted. In addition, we consider the genes that appear as input in a linear regression if its estimated error is smaller than a threshold value. Then the edge defined by the output and input genes is added to the graph. Otherwise, the gene is not considered in the graph, and hence it is not in the gene network either.

4 Experiments

We use the Rossetta compendium to evaluate the results obtained and to analyze if a gene network is biologically coherent. The Rosetta compendium consists of 300 full–genome expression profiles of the yeast Saccharomyces cerevisiae. Therefore, we work with a dataset with 300 examples and 6316 genes.

We use the iron homeostasis pathway in yeast (which regulate the uptake, storage, and utilization of iron so as to keep it at a non–toxic level) to evaluate the accuracy of our model. The most important genes in this process are: FRE1, FRE2, FTR1 and FET3, which control the reductive mechanism; ARN1, ARN2, ARN3 and ARN4, which control the non reductive mechanism and, finally, FIT1, FIT2 y FTI3 which are connected in the iron transport. Relevant set of genes have been taken from [21]. This set of genes has been used to measure the accuracy of bayesian gene networks in previous projects, as in [22], [23] and [20]. Specifically, those papers show models of the iron homeostasis pathway from the Rossetta compendium and are centered on ARN1 gene. Therefore, we study our approach starting at ARN1 gene in the first iteration.

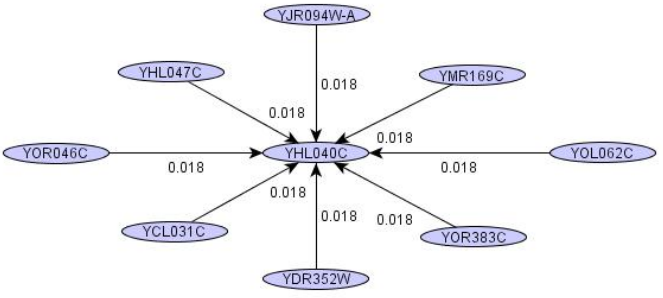


Fig. 1. An example of gene network generated using a seed gene (ARN1) and only one iteration for the algorithm. We weight the edges of the network. The weight is calculated by: $weight = \frac{1}{error \times frequency}$ where *error* is the relative absolute error and *frequency* is the number of times that a gene is in the regression tree.

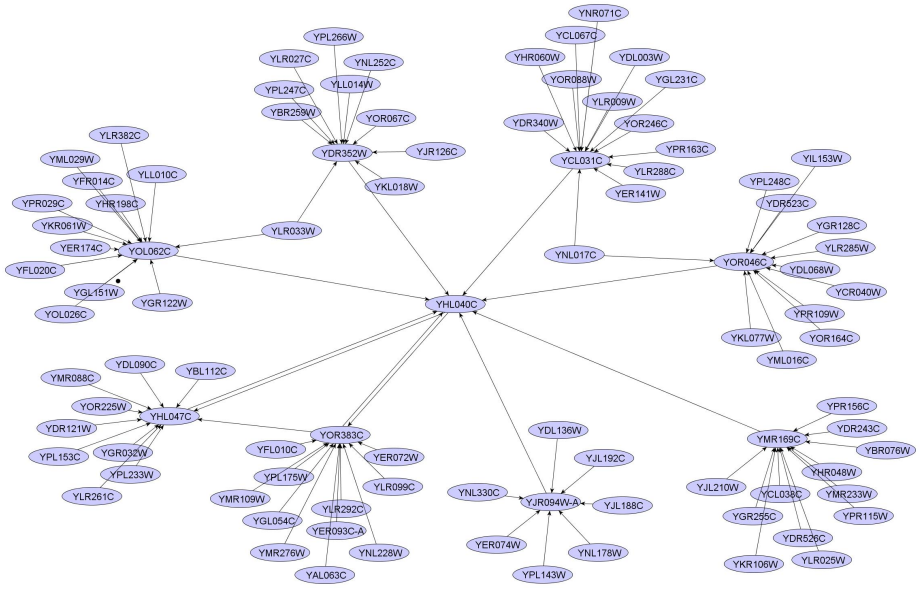


Fig. 2. Gene network for the iron homeostasis process, starting with the gene ARN1 and allowing up to 12% of the instances, i.e. each leaf of the regression trees have to cover al least 12% of the total number of instances in the dataset

The algorithm starts using as a seed the gene ARN1 (ORF YHL040C). In order to control the accuracy of each leaf, each of them has to cover at least 33% of the total number of instances in the dataset. In the first iteration (distance 1 from the seed gene), the algorithm finds the gene FIT3 (ORF YOR383C) (see Figure 1) whereas the next gene is found in the fourth iteration. If the constraint is relaxed (allowing up to 12% of the instances), the algorithm finds in the first

iteration two genes related to the iron homeostasis: FIT3 (ORF YOR383C) and ARN2 (ORF YHL047C). The gene network is depicted in Figure 2.

In the experiment above we run the algorithm from a seed gene and a gene network is built step by step from that seed gene. This is an iterative method where the number of iterations must be calculated by experimentation. Now, we study the results of the algorithm studying the whole regression trees generated for each gene. Evaluating the results is a difficult task because of the huge network generated taking into account all the genes of the dataset. Therefore the structure of the network is analyzed considering, in the task of build the network, only those regression trees whose relative absolute error is lower than a threshold value together with the branches of the trees whose support is greater than a threshold value. These two thresholds are studied by experimentation as we can see in Figure 3. The top-left figure shows the size of the network as a function of the relative absolute error. Each line is obtained by setting the value of the other threshold as 0%, 25%, 50%, 75% and 100%. We can observe that the size of the network increases slowly for an absolute relative error higher than 60. The top-right figure shows the size of the network as a function of the support. Each line is obtained by setting the value of the error as 0%, 25%, 50%, 75%

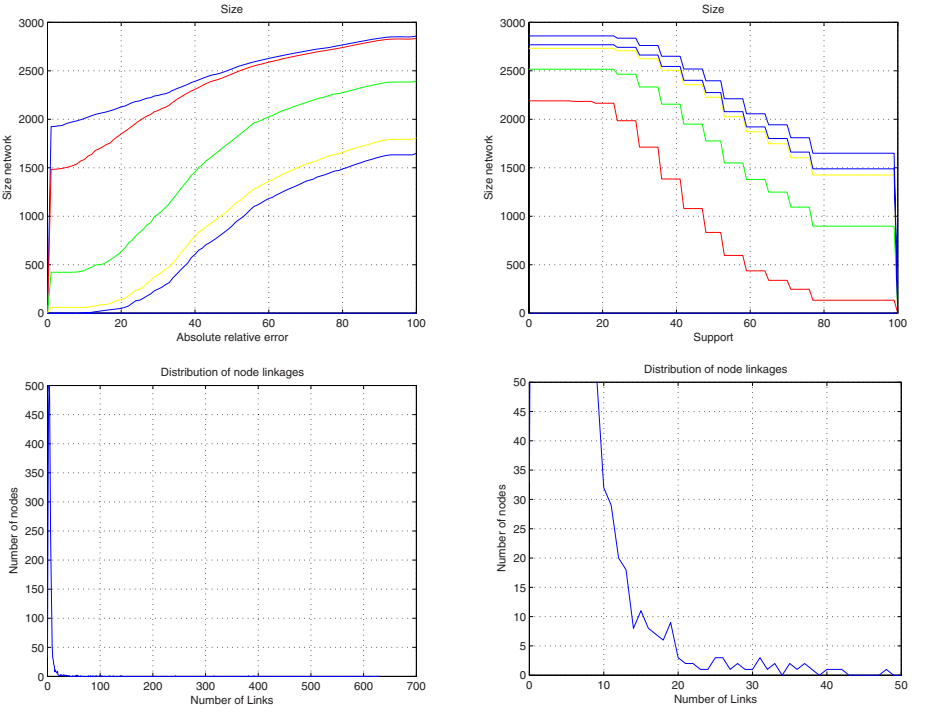


Fig. 3. The top figures show the size of the networks as a function of the relative absolute error and support and the bottom figures show the distribution of node linkages

and 100%. In the bottom-left and bottom-right figures, we depict the distribution of node linkages. The bottom-right figure is another representation of the bottom-left one, in which we can better appreciate that it follows a power law in that most nodes have just a few connections and only a few have a huge number of links.

5 Conclusions

In this work we present a method to build gene networks from regression trees. The approach has several advantages: first, the gene network is based on the accuracy and support of the regression trees, which leads to solutions driven by the user; second, the gene–gene interactions are based on the local regressions, much more accurate than global regressions; third, many possible relationships are initially pruned if they do not fit the thresholds; and fourth, the depth and size of the graph is controlled by the number of iterations or the number of the studied genes, so some relationships among genes are discovered indirectly, as they might not be directly related.

The approach has an accurate performance, although we are working on designing better pruning techniques, focused on selecting only relevant genes from the regression trees to be further included in the gene network. Secondly, we are working on preparing an experimental study in which the performance of our approach is compared with other methods. And finally, we are working on determining the number of iterations and the threshold values of our approach.

Results presented in this paper, using only one gene as a seed to build the gene network are very promising, as the gene network involves genes that might have interesting biological properties for the process in study.

References

1. Tavazoie, S., Hughes, J., Campbell, M., Cho, R., Church, G.: Systematic determination of genetic network architecture. Nature Genetics 22, 281–285 (1999)
2. Silvescu, A., Honavar, V.: Temporal Boolean Network Models of Genetic Networks and Their Inference from Gene Expression Time Series. Complex Systems 13, 54–70 (2001)
3. Chen, K., Calzone, L., Csikasz-Nagy, A., Cross, F., Novak, B., Tyson, J.: Integrative Analysis of Cell Cycle Control in Budding Yeast. Molecular Biology of the Cell 15, 3841–3862 (2004)
4. Matsuno, H., Doi, A., Nagasaki, M., Miyano, S.: Hybrid Petri net representation of gene regulatory network. Pacific Symposium on Biocomputing 5, 87 (2000)
5. Shmulevich, I., Dougherty, E., Zhang, W.: Gene perturbation and intervention in probabilistic Boolean networks. Bioinformatics 18, 1319–1331 (2002)
6. Stuart, J.M., Segal, E., Koller, D., Kim, S.K.: A gene-coexpression network for global discovery of conserved genetic modules. Science 302, 249–255 (2003)
7. Magwene, P.M., Kim, J.: Estimating genomic coexpression networks using first-order conditional independence. Genome Biol. 5 (2004)

8. Basso, K., Margolin, A., Stolovitzky, G., Klein, U., Dalla-Favera, R., Califano, A.: Reverse engineering of regulatory networks in human B cells. Nature Genetics 37, 382–390 (2005)
9. Friedman, N.: Using bayesian networks to analyze expression data. Journal of Computational Biology 7, 601–620 (2001)
10. Quinlan, J.: Learning with continuous classes. In: Procedings Australian Joint Conference on Artificial Intelligence, pp. 343–348 (1992)
11. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Technical report, Morgan Kaufmann, San Mateo (2000)
12. Karalic, A.: Linear regression in regression tree leaves. In: Proceedings of the ISSEK 1992, Bled, Slovenia (1992)
13. Torgo, L.: Functional models for regression tree leaves. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 385–393 (1997)
14. Torgo, L.: Partial linear trees. In: ICML 2000. Proceedings of the 17th International Conference on Machine Learning, pp. 1007–1014 (2000)
15. Li, K., Lue, H., Chen, C.: Interactive Tree-Structured Regression Via Principal Hessian Directions. Journal of the American Statistical Association 95 (2000)
16. Battle, A., Segal, E., Koller, D.: Probabilistic discovery of overlapping cell processes and their regulation. In: RECOMB. Eight Annual International Conference on Research in Computational Molecular Biology, San Diego, CA (2004)
17. Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D.: Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nature Genetics 34, 166–176 (2003)
18. Hughes, T., Marton, M., Jones, A., Roberts, C., Stoughton, R., Armour, C., Bennett, H., Coffey, E., Dai, H., He, Y.: Functional Discovery via a Compendium of Expression Profiles. Cell 102, 109–126 (2002)
19. Fink, G., Spellman, P., Sherlock, G., Zhang, M., Iyer, V., Anders, K., Eisen, M., Brown, P., Botstein, D., Futcher, B.: Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization. Molecular Biology of the Cell 9, 3273–3297 (1998)
20. Peña, J., Bjorkegren, J., Tegner, J.: Discovering statistically significant biclusters in gene expression data. Bioinformatics (in press)
21. Lesuisse, E., Blaiseau, P., Dancis, A., Camadro, J.: Siderophore uptake and use by the yeast Saccharomyces cerevisiae (2001)
22. Margolin, A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., Califano, A.: Aracne:an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. BMC Bioinformatics 7, S7 (2006)
23. Peer, D., Regev, A., Elidan, G., Friedman, N.: Inferring Subnetworks from Perturbed Expression Profiles. Bioinformatics 17, 215–224 (2002)

Profiling of High-Throughput Mass Spectrometry Data for Ovarian Cancer Detection

Shan He and Xiaoli Li

Cercia, School of Computer Science
The University of Birmingham, Edgbaston, Birmingham, B15 2TT, U.K.
s.he@cs.bham.ac.uk

Abstract. Mass Spectrometry (MS) has been applied to the early detection of ovarian cancer. To date, most of the studies concentrated on the so-called whole-spectrum approach, which treats each point in the spectrum as a separate test, due to its better accuracy than the profiling approach. However, the whole-spectrum approach does not guarantee biologically meaningful results and is difficult for biological interpretation and clinical application. Therefore, to develop an accurate profiling technique for early detection of ovarian cancer is required. This paper proposes a novel profiling method for high-resolution ovarian cancer MS data by integrating the Smoothed Nonlinear Energy Operator (SNEO), correlation-based peak selection and Random Forest classifier. In order to evaluate the performance of this novel method without bias, we employed randomization techniques by dividing the data set into testing set and training set to test the whole procedure for many times over. Test results show that the method can find a parsimonious set of biologically meaningful biomarkers with better accuracy than other methods.

1 Introduction

While ovarian cancer accounts for fewer deaths than breast cancer, it still represents 4% of all female cancers. Moreover, ovarian cancer is rarely detected in early stage and also particularly aggressive: when detected in late stages, e.g., stage III and beyond, the 5-year suvival rate is approximately 15% [1]. Detection of early-stage ovarian cancer can reduce the death rate significantly. For example, the reported 5-year suvival rate is about 90% for those women deteted in stage I. Cancer antigen 125 (CA125) has been introduced for cancer diagnosis [2]. However, the accuracy for early-stage cancer diagnosis is very low (about 10%) and is prone to large false positive rate.

Recently, Mass Spectrometry as a proteomics tool is applied to early-stage cancer diagnosis. This new proteomics tool is simple, inexpensive and minimally invasive [3]. The first application of MS to the early-stage cancer diagnosis was done by Petricoin [4] on ovarian cancer. The author employed genetic algorithms (GAs) coupled with clustering analysis to generate diagnosis rule sets to predict ovarian cancer. The study was based on the SELDI-TOF (Surface-enhanced

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 860–869, 2007.

Laser Desorption/Ionzation Time-Of-Flight) low-resolution MS data. With the advance of the mass spectrometry technology, high-resolution SELDI-TOF was employed and studied by the same authors to discriminate ovarian cancer from normal tissue. This dataset is collected with extensive quality control and assurance (QC/QA) analysis which are supposed to have superior classification patters when compared to those collected with low-resolution instrumentation [1]. In their paper, the sensitivity and specificity were claimed to be both almost 100%. However, a reproducing study done by Jerries [5] shows that the performance of the best prediction model generated by their GA only achieved 88% accuracy at 25th percentile and 93% accuracy at 75th percentile.

Recently, in attempt to improve the accuracy of identifying cancer on the high-resolution SELDI-TOF ovarian cancer data, Yu et.al. [6] proposed a method that consists of Kolmogorov-Smirnov (KS) test, wavelet analysis and Support Vector Machine (SVM). The average sensitivity and specificity are 97.38% and 93.30%. Before the classification using SVM, the proposed method selected 8094 m/z values via KS test and further compressed to a 3382-dimensional vector of approximation coefficients with Discrete Wavelet Transformation (DWT). Although the accuracy achieved by the procedure was improved, the biological interpretability was greatly sacrificed since the 3382-dimensional DWT coefficient vector for classsifcation is not biologically meaningful. In fact, our recent research [7] shows that, by sacrificing biological interpretability, simple Principle Component Analysis (PCA) coupled with Linear Discriminant Analysis (LDA) [8] can achieved averaged sensitivity of 98.4633% and an average specificity of 97.0730% in 1000 independent k-fold cross validation test, where $k = 2, \ldots, 10$, which is better than the results obtained by [6] with less computational overhead.

In [9], current methods of distinguishing cancer and control groups based on the SELDI-TOF MS data can be classified as 1). taking a list of peak found in cross spectra as input; 2). treating entire spectra as input and attempt to identify m/z values that serve as biomarkers. In this paper, we refer the first category as "profiling" method and the second one as "whole-spectrum" method. The authors [9] argued that the profiling method is more important since it guarantees that the features, in this case, peaks, are "more biologically meaningful in that they represent chemical species that can be subsequently identified and studied".

In this paper, we propose a novel profiling method for ovarian cancer identification on the high-resolution SELDI-TOF data. The aim of the paper is to propose a general method can generate more accurate, and also biologically meaningful results. The proposed method therefore could serve as a diagnostic tool and a biomarker discovery tool, which is of great importance to physicians and pharmacy industry. Thus, we argue that peak detection is the most important step. The reason is that, a successful peak detection algorithm is capable to detect most of true peaks with minimum false peak detections, which greatly reduce the complexity of feature selection, consequently increase the accuracy of classification. In this paper, in the first time, we introduce Smoothed Nonlinear

Energy Operator (SNEO), which has been successfully used in EEG signal processing for spike detection, to the peak detection of SELDI-TOF data. To reduce the number of peaks detected, important peaks are selected using a filter based feature selection method, correlation-based feature selection. The selected peak set is then used to build a classification model using a Random Forest classifier. In order to investigate the importance of each biomarker to the identification of ovarian cancer, we utilize the built-in variable selection feature of Random Forest classifier to calculate the variable importance value of each biomarker so the obtained biomarkers can be ranked.

2 Methods

As shown in Figure 1, the method consists of the following major steps: (1). data preprocessing; (2). SNEO based peak detection; (3). peak calibration; (4). correlation-based peak selection; (5). peak extraction; (6). Random Forest (RF) based classification. In the following sections, we give details of each step.

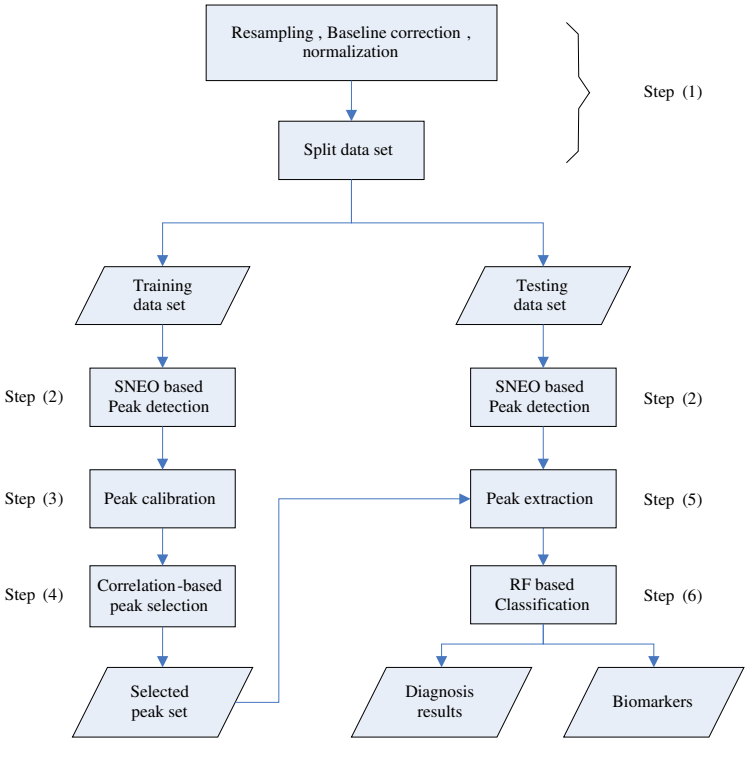


Fig. 1. The proposed method for biomarker discovery

2.1 Data Preprocessing

The high-resolution MS dataset was collected from a hybrid quadrupole time-of-flight mass spectrometer (QSTAR pulsar I, Applied Biosystems, Inc. Framingham, MA, USA) with WCX2 ProteinChip. Detailed information of the data collection can be found in [1]. Each original spectrum possesses approximately 350000 m/z data points, which requires intensive computational cost. Moreover, the data points of each original spectrum are also different. In order to compare different spectra under the same reference and at the same resolution, it is necessary to homogenize the m/z vector. We employ a resampling algorithm in the MATLAB Bioinformatics Toolbox to resample the data to 7084 m/z points per spectrum.

We correct the baseline caused by the chemical noise in the matrix or by ion overloading using the following procedure: 1). estimated the baseline by calculating the minimum value within the width of 50 m/z points for the shifting window and a step size of 50 m/z points; 2). regresses the varying baseline to the window points using a spline approximation; and 3). subtract the resulting baseline from the spectrum. Finally, each spectrum was normalized by standardizing the area under the curve (AUC) to the median of the whole set of spectrum. The data set is split for training and testing as detailed in Section 3.

2.2 SNEO Based Peak Detection

Smoothed Non-linear Energy Operator (SNEO), or also known as the Smoothed Teager Energy Operator, has been used to detected hidden spikes in EEG and ECG biomedical signal. The method is sensitive to any discontinuity in the signal. It was shown by [10] that the output of SNEO is the instantaneous energy of the high-pass filtered version of a signal. For MS data, true peaks can be regarded as instantaneous changes in the signal. Therefore, the SNEO is ideal for the detection peaks in MS data because of its instantaneous nature. The generalized SNEO Ψ_s is defined as [10]:

$$\Psi_s[x(n)] = \Psi[x(n)] \otimes w(n) \tag{1}$$

$$\Psi[x(n)] = x^2(n) - x(n+j)x(n-j) \tag{2}$$

where $\otimes$ is the convolution operator and $w(n)$ is a smoothing window function; in this study, bartlett window function is used. Usually, the step size j is set to be 1 which gives us a standard SNEO. For the high-solution MS data, we selected the step size $j = 3$, which gives the best classification results.

After applying SNEO to pre-emphasis peaks in the signal, potential peaks are detected by a threshold. An optimal threshold is to minimize the missing of true peaks, while keeping the number of false peaks within a reasonable limit [10]. In [10], a scaled version of the mean of the SNEO output is defined as the threshold:

$$\tau = C\frac{1}{N}\sum_{n=1}^{N}\Psi_s[x(n)] \tag{3}$$

where C is the scaling factor and N is the number of samples. In this study, $C = 0.1$, which generates the best classification results.

2.3 Peak Calibration

The correlation-based feature selection [11] uses a correlation based heuristic to determine the usefulness of feature subsets. The usefulness is determined by measuring the "merit" of each individual feature for predicting the class label as well as the level of intercorrelation among them. First, an evaluation function is defined as:

$$G_s = \frac{k\overline{r_{ci}}}{\sqrt{k + k(k-1)\overline{r_{ii}}}} \tag{4}$$

k is the number of features in the subset; $\overline{r_{ci}}$ is the mean feature correlation with the class, and $\overline{r_{ii}}$ is the average feature intercorrelation.

Equation (4) is the core of the feature selection algorithm. With this evaluation function, heuristic search algorithm then can be applied to search the feature subset with the best merit as measured in Equation (4).

In order to measure the correlation between features and the class (r_{ii}), and between features (r_{ci}), there exist broadly two approaches. One is based on classical linear correlation and the other is based on information theory. The correlation based feature selection employed the information theory based approach since it can capture the correlations that are not linear in nature. The following equations give the entropy of Y before and after observing X

$$H(Y) = -\sum_{y \in R_y} p(y)\log(p(y)) \tag{5}$$

$$H(Y|X) = -\sum_{x \in R_x} p(x) \sum_{y \in R_y} p(y|x)\log(p(y|x)) \tag{6}$$

Based on the measurement of correlation of Y on X, Uncertainty coefficient of Y is calculated [11]

$$C(Y|X) = \frac{H(Y) - H(Y|X)}{H(Y)} \tag{7}$$

The output of $C(Y|X)$ lies between 0 (X and Y have no association) and 1 (knowledge of X completely predicts Y). From Equation (5), (6) and (7), r_{ii} and r_{ii} can be calculated.

It can be shown that this uncertainty coefficient is actually derived from mutual information. The mutual information $I(Y, X)$ between Y and X is defined as

$$I(Y, X) = H(Y) + H(X) - H(Y, X) \tag{8}$$

where H(Y,X) is the joint entropy which can be expressed in terms of the conditional entropy $H(Y|X)$

$$H(Y, X) = H(Y|X) + H(X) \tag{9}$$

Therefore, $I(Y, X)$ can be written as

$$\begin{aligned} I(Y, X) &= H(Y) + H(X) - H(Y, X) \\ &= H(Y) + H(X) - (H(Y|X) + H(X)) \\ &= H(Y) - H(Y|X) \end{aligned}$$

It is obvious that

$$C(Y|X) = \frac{I(Y,X)}{H(Y)}.$$

2.4 Peak Extraction

After applying the correlation-based peak selection to the detected peak from the training data set, a small set of peaks then can be generated. This set of peaks will be used as inputs for the Random Forest classifier to build a prediction model. Based on the selected peak set, we construct m/z window using the same width ($N = 6$) as used in the calibration step. Peaks detected by SNEO peak detection algorithm from the testing data set is extracted by the constructed m/z window, that is, only those peaks within the m/z window will be used as inputs in the testing.

2.5 Random Forest Based Classification

Random Forest (RF) classifier [12] consists of many unpruned decision trees and outputs the class that is the mode of the classes output by individual trees [12]. The basic idea behind this classifier is combining "bagging" techniques, that is, bootstrap aggregation and random variable selection for tree building to construct a collection of decision trees with controlled variations. The random forest classifier can be defined as a ensemble of K classifiers $h_1(\mathbf{x})$, $h_2(\mathbf{x})$, $\cdots$, $h_K(\mathbf{x})$, where attribute vector $\mathbf{x}$ consists of attributes. The pseudo-code is described in Table 1.

In recent years, the RF classifier is gaining popularity due to the following advantages:

- It produces high accuracy for many problems without over-fitting;
- It is a fast algorithm, even faster than growing and pruning a single tree. Moreover, it is also inherently parallelable;
- It can handle high dimensional input variables without much problem;

Table 1. Pseudo-code for the Random Forest Classifier

Set $k = 1$;
Bootstrap sample observations
FOR (each sample i in bootstrapped dataset)
Grow an unpruned classification tree $h_k(\mathbf{x})$ for sample i
FOR (each node in the classification tree $h_k(\mathbf{x})$
Randomly sample k of the predictor variables
Choose the best split from among those variables
END FOR
END FOR
Predict new data by combining the predictions of the trees
Calculate summary statistics and variable importance, etc.

– It particularly easy to use because there is only one tuning parameter.
– It estimates the importance of input variables in determining classification.

Among the advantages, the last one is the most interesting and useful for this study, which can be used to perform variable selection. It is done by measuring the decrease of classification accuracy when values of variable in a node are permuted randomly [12]. In this study, we utilize this feature to calculate the importance of each biomarker.

The only tuning parameter is the number of variables randomly sampled as candidates at each split. In this study, we set the number to be 1.

3 Results

In order to evaluate the performance of the proposed method without bias, we split the 216 samples, of which 95 control and 121 cancer, into training and testing data sets. 52 control samples and 53 cancer samples were selected for training data. The rest 43 control samples and 68 cancer samples were set aside for evaluation as a blind data set. This same setting was used in [1] and [5].

We compared the performance of the proposed method, especially the performance of SNEO peak detection, with two commonly used peak detection method. The first one was proposed by Yasui et. al. in [13]. In this method, a peak is detected if it takes the maximum value in the k-nearest neighborhood. The selection of k is critical to the performance of Yasui's peak detection algorithm. In [13], it was done by trial and error with visual checking of the resulting peak/non-peak data. In this study, $k = 20$, which results the best accuracy in the classification step, is used. The second method is Cromwell package which was proposed by Coombes et. al. in [14]. This first step of this method is denoising the MS spectra with Undecimated Discrete Wavelet Transform (UDWT). Baseline correction and normalization are then applied. Peaks are detected by locating maxima in each proposed spectrum and then are consequently qualified with Signal-to-Noise ratios. Finally, the detected peaks are calibrated by combining peaks that differed in location by no more than 7 clock ticks.

In order to compare these peak detection algorithms with our SNEO based detection algorithm without bias, we replaced the SNEO peak detection algorithm with these two algorithms and keep the rest steps unchanged, then applied the three methods to the same randomly split data set. Based on the different peak detection methods, we termed these three methods as SNEO, Yasui, and Cromwell, respectively.

We repeated the whole procedure of the three methods for 1000 times and calculated the average results as listed in Table 2. It can be found from the table that, the overall test set accuracy generated by our proposed method (SNEO) at 25 and 75 percentiles are all better than the Yasui and Cromwell's methods.

In comparison with the original publication on the same ovarian data set [1], which claimed of average 100% sensitivity and specificity, the prposed method only achieved average 94.49% and 94.68% sensitivity and specificity. However,

Table 2. Test set accuracy percentiles from 1000 runs

Algorithm	Test set accuracy 25th			Test set accuracy 75th		
	Overall	Sensitivity	Specificity	Overall	Sensitivity	Specificity
SNEO	92.79	92.72	92.72	96.39	98.16	98.11
Yasui	89.18	93.10	83.67	93.69	98.14	92.15
Cromwell	87.39	91.66	81.81	91.89	98.21	89.89

in [1], Jerries reproduced the method using a GA on the same data set. The overall accuracy of the GA was only 88% and 93% at 25 and 75 percentiles, respectively, which is far less than the results obtained by our prposed method. Apart from its poorer accuracy, the GA based method actually falls into the whole-spectrum method as stated in Section 1, since the method treated each point in the spectrum as a specarate test. The outputed biomarkers are a set of significant m/z values but are not necessary a peak set. Therefore, the biological interpreation of their results is not guanranteed.

In [6], the average sensitivity and specificity were improved to 97.38% and 93.30% in 1000 independent k-fold cross-validation, where $k = 2, \cdots, 10$. However, by applying KS test and DWT to reduce dimensionality of the data, the biological iterpretablility of this mehtod is greatly sacrisfied. The features used for classification were a set of coefficients of DWT, which are even less biologically meaningful than a set of m/z values.

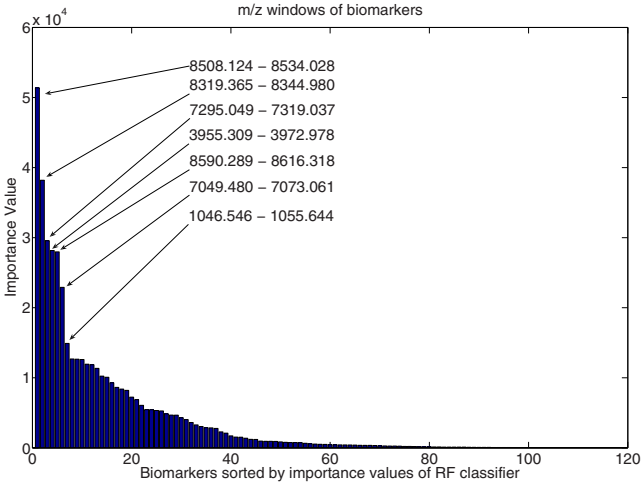


Fig. 2. Biomarkers sorted by feature importance values generated by Random Forest (RF) classifier. It is calculated by summing up each biomarker's importance value that contribute to the classification of RF classifier over 1000 runs. The higher importance value, more important the biomarker contributes to the classification.

In each run, our proposed method selected approximately 10 biomarkers on average. In order to understand how important these biomarkers contribute to the identification of ovarian cancer, we calculated the sum of importance values of each biomarker in the 1000 runs by fully exploiting the variable selection feature of RF classifier. As shown in Figure 2, 108 biomarkers found during the 1000 runs are ranked by their importance values and the most important 7 biomakers are labeled. It is interesting to note that, in [1], two common m/z values recurring in their four distinct models, 7060.121 and 8605.678, fall into the m/z windows of the most important biomarkers discovered by our method.

4 Conclusion

In this study, we propose a novel profiling method for high-resolution MS data of ovarian cancer. The core of the method is the SNEO peak detection algorithm, which is introduced from EEG/ECG signal processing literature to MS data analysis for the first time. Correlation-based peak selection is employed to select a parsimonious peak set that generates the most accurate classification results. RF classifier is then applied to identify ovarian cancer based on the selected peak set.

We evaluated the proposed method by using the same experimental settings used in [1] and [5]. Results from our method are better than the method presented in [5]. Besides its good accuracy, compared to the whole-spectrum methods e.g., the method of [6], the most notable merit of our proposed method is that it obtains more biologically meaningful results for further study and validation. We also compared our method with another two methods based on two popular peak detection algorithms, namely, Yasui's peak detection algorithm and the Cromwell peak detection algorithm. The proposed SNEO-based method also markedly outperformed these two methods in terms of accuracy.

Acknowledgement. This work is supported by the Leverhulme Trust.

References

[1] Conrads, T., Fusaro, V., Ross, S., Johann, D., Rajapakse, V., Hitt, B., Steinberg, S., Kohn, E., Fishman, D.: High-resolution serum proteomic features for ovarian cancer detection. Endocr. Relat. Cancer 11(2), 163–178 (2004)

[2] Zurawski, V.R., Orjaseter, H., Andersen, A., Jellum, E.: Elevated serum ca 125 levels prior to diagnosis of ovarian neoplasia: relevance for early detection of ovarian cancer. Int. J. Cancer 42(5), 677–680 (1988)

[3] Zhang, X., Wei, D., Yap, Y., Li, L., Guo, S., Chen, F.: Mass spectrometry-based "omics" technologies in cancer diagnostics. Mass Spectrometry Reviews 26, 403–431 (2007)

[4] Petricoin, E., Ardekani, A., Hitt, B., Levine, P., Fusaro, V., Steinberg, S., Mills, G., Simone, C., Fishman, D., Kohn, E.: Use of proteomic patterns in serum to identify ovarian cancer. The Lancet 359, 572–577 (2002)

[5] Jeffries, N.O.: Performance of a genetic algorithm for mass spectrometry proteomics. BMC Bioinformatics 5(1), 180 (2004)

[6] Yu, J.S., Ongarello, S., Fieldler, R., Chen, X.W., Toffolo, G., Cobelli, C., Trajanoski, Z.: Ovarian cancer identification based on dimensionality reduction for high-throughput mass spectrometry data. Bioinformatics 21(10), 2200–2209 (2005)
[7] He, S., Li, X.L.: Profiling of high-throughput ovarian cancer mass spectrometry data using smoothed non-linear energy operator and random forest - preliminary results. Technical report, CERCIA, School of Computer Science, the University of Birmingham (2007)
[8] Li, X.L., Li, J., Yao, X.: A wavelet-based pre-processing technique for mass spectrometry. Computers in Biology and Medicine 37, 509–516 (2007)
[9] Carlson, S.M., Najmi, A., Whitin, J.C., Cohen, H.J.: Improving feature detection and analysis of surface-enhanced laser desorption/ionization-time of flight mass spectra. Proteomics 5(11), 2778–2788 (2005)
[10] Mukhopadhyay, S., Ray, G.: A new interpretation of nonlinear energy operator and its efficacy in spike detection. IEEE Transactions on Biomedical Engineering 45(2), 180–187 (1998)
[11] Hall, M.A.: Correlation-based Feature Selection for Machine Learning. PhD thesis, The University of Waikato (1999)
[12] Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
[13] Yasui, Y., Pepe, M., Thompson, M.L., Adam, B.L., Wright, G.L., Qu, Y., Potter, J.D., Winget, M., Thornquist, M., Feng, Z.: A data-analytic strategy for protein biomarker discovery: profiling of high-dimensional proteomic data for cancer detection. Biostatistics 4(3), 449–463 (2003)
[14] Coombes, K.R., Tsavachidis, S., Morris, J.S., Baggerly, K.A., Hung, M.C., Kuerer, H.M.: Improved peak detection and quantification of mass spectrometry data acquired from surface-enhanced laser desorption and ionization by denoising spectra with the undecimated discrete wavelet transform. Proteomics 5(16), 4107–4117 (2005)

Adapting Machine Learning Technique for Periodicity Detection in Nucleosomal Locations in Sequences

Faraz Rasheed[1], Mohammed Alshalalfa[1], and Reda Alhajj[1,2]

[1] Computer Science Dept, University of Calgary, Calgary, Alberta, Canada
[2] Department of Computer Science, Global University, Beirut, Lebanon

Abstract. DNA sequence is an important determinant of the positioning, stability, and activity of nucleosome, yet the molecular basis of these remains elusive. Positioned nucleosomes are believed to play an important role in transcriptional regulation and for the organization of chromatin in cell nuclei. After completing the genome project of many organisms, sequence mining received considerable and increasing attention. Many works devoted a lot of effort to detect the periodicity in DNA sequences, namely, the DNA segments that wrap the Histone protein. In this paper, we describe and apply a dynamic periodicity detection algorithm to discover periodicity in DNA sequences. Our algorithm is based on suffix tree as the underlying data structure. The proposed approach considers the periodicity of alternative substrings, in addition to considering dynamic window to detect the periodicity of certain instances of substrings. We demonstrate the applicability and effectiveness of the proposed approach by reporting test results on three data sets.

1 Introduction

Eukaryotic DNA is complex with basic histone proteins forming nucleosome and higher order chromatin. The ability of the histone octamer to wrap differing DNA sequences into nucleosomes is highly dependent on the specific DNA sequence. Those sequence fragments that wrap Histone proteins got a lot of attention because it is of great importance to determine the positioning of nucleosome which affects gene regulation positively and negatively.

In this work we consider DNA sequence as a time series data. Time series is a sequence of data values collected usually at a uniform time interval. Real-life applications have several examples like number of transactions per hour in a superstore, hourly weather data for particular location, etc. We detect the periodicity of certain sequences (dinucleotides) within a given DNA sequence. Periodicity mining in time series data and in DNA sequence in particular is a hot area of research. Our approach employs suffix trees as the basic data structure; we map the DNA sequence into suffix tree based representation, which is then used to find the periodicity in the DNA sequence.

Given the sequence of many DNA segments, the problem is to find the periodicity of certain sequences along the whole DNA sequences. Periodicity detection in biological data is different in many ways from the traditional time series data. First,

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 870–879, 2007.

there is the concept of alternatives where a group of strings can replace each other. The task is to find the periodicity of a group by considering alternative strings. For instance, in our experiments the strings TA, TT and AA are parts of an alternative group, and the presence of any of these is counted as valid repetition. Hence, the sequence TTACGAATGGTAGT has the periodicity for alternative string group (AA, TT, TA) with period = 5 starting at position 0 and periodic strength = 1 (or 100%).

Secondly, we introduce the concept of a window for detecting periodic occurrences. In traditional time series data, we expect to find the periodic string (or symbol) at positions *stPos*, *stPos*+*p*, *stPos*+*2p*, and so on, where *stPos* is the starting position of the periodic pattern while *p* is period value. For example, in the sequence abdadbacc, 'a' is periodic with period $p = 3$ starting from $stPos = 0$ with periodic strength of 100%. Hence, we expect to find 'a' to appear at index positions 0, 3, 6, …, which are *stPos*, *stPos*+*p*, *stPos*+*2p*, …. Unfortunately, DNA sequences generally do not repeat that strictly. For example, in the chicken nucleosomal sequence that we used in our experiments, the alternative string group (AA, TT, TA) is expected to be repeated with the periodicity of 10, but it is not guaranteed that the alternative string group would repeat after every 10 index positions; rather the alternative string group may appear ±2 positions away from the expected periodic difference of 10. This means that if the first appearance of the alternative group (say TT) is found at position 6 and the expected period is 10, then the second appearance of the alternative group (say TA) may be found from position 14 to position 18 (16±2), if the relaxed occurrence range (or window) is taken as 2. Formally, the appearance of the occurrence string is considered as valid if it is found at index position *stPos*, $stPos + p \pm RR$, $stPos + 2p \pm RR$, $stPos \pm 3p + RR$, …, where *RR* is the relaxed range window under which the occurrence would be considered valid.

Thirdly, when applying periodicity detection algorithm on DNA sequence, we are only interested in the periodicity of a specific string (alternative string group) and not in the periodicity detection of all substrings and symbols. Further, we are only interested in a specific period range and not in all the periods.

While achieving the third requirement might be possible with some of the periodicity detection algorithms with few adjustments, most of them, especially the ones which work on the shift & compare principle, e.g., [1, 2], cannot meet the first two requirements. Our algorithm is flexible enough to fulfill these requirements as it uses suffix tree and keeps the occurrence vector for each repeating sub-sequence.

Suffix tree based data structure allows us to quickly find all the occurrences of all repeating sub-sequences (sub-strings) in linear time. Since we have the separate occurrence vector for each sub-sequence, we may combine the occurrences of each string in the alternative string group into one vector. As our algorithm calculates the periodicity using the occurrence vector of a sub-string, we can run the algorithm for periodicity calculation on the combined occurrence vector of alternative string group. Since we validate the repetition by taking the mod of occurrence position with starting position (as can be seen in the algorithm presented in Figure 3), we can also accommodate the relaxed range occurrence concept by simply checking if a particular occurrence falls in the relaxed range window of the expected periodic position.

The rest of the paper is organized as follows. Related work is presented in Section 2. Section 3 contains the description of the approach. Results and discussion is presented in Section 4. Section 5 is conclusions.

2 Related Work

Periodicity detection both in time series databases, e.g., [2] and in DNA sequences, e.g., [18] is getting more active and more significant. As finding periodicity in time series databases in concerned, Indyk *et al* [1] presented their periodic trends algorithm which can find the segment periodicity only, i.e., to detect if the entire time series can be achieved by the repetition of a sequence of symbols. Elfeky *et al* [2] presented two algorithms to find symbol and segment periodicities in the time series, but their algorithm favors shorter periods. On the other hand, our algorithm is capable of detecting symbol, segment, and sequence periodicity. The algorithm does not favor any period size. Unlike the work of Elfeky, we do not need two separate algorithms to find the symbole and segment periodicities. Our single algorithm can detect segments, symbol and sequence periodicities by just a single pass over the data once the series is represented in suffix tree.

To the best of our knowledge, no existing algorithm of periodicity detection can fulfill all the three requirements outlined in the previous section. We elaborate on this by comparing our algorithm with the three well-known algorithms, namely, Indyk [1], Elfeky [2,14], and Ma [15] for the adaptablity to work with biological data.

For the first requirement, the concept of alternatives where a group of strings can replace each other, Indyk [1] and Elfeky [14] can not be applied as they find the segment periodicity of entire time series and not the for a single or a sequence of alphabets. Elfeky [2] and Ma [15] can find the symbol periodicity, which is the periodicity of a single symbol, but can not find the periodicity of a sequence of symbols. So, if we wish to use this algorithm to find the periodicity of 'TA', we need to calculate the periodicity of 'T' and 'A' separately, and then combine the periodicity results using Han's approach [16]; still it won't give the exact periodic strength of 'TA', rather it would give the (min-max) range for the periodic strength. If we wish to implement the alternative strings concept in [2], we need to replace all occurrences of each string in alternative strings group by a single string and then we can run the algorithm to find the periodicity of that string. Clearly, there is disadvantage in this approach as we have to make a copy of the entire time series for each alternative string group.

For the second requirement, the concept of time tolerance (or relaxed range) window for periodic occurrences, it is used to accommodate various types of noise (insertion, deletion, replacement or a mixture of these) in the data. Since Indyk [1] does not calculate the symbol (or sequence) periodicity, it can not be used in this situation. Elfeky [2] algorithm can not accommodate the time tolerance window concept as it shifts the entire time series for each candidate period. Recently, Elfeky *et al* presented a different algorithm called 'WARP' [14] to deal with this problem. But their new algorithm can only find segment periodicity (periodicity for the entire time series) and not symbol periodicity. Even if we somehow generate the DTW matrix for each alternative string, it would worsen the complexity of the algorithm (which is already $O(n^2)$ compared to their convolution algorithm's [2] complexity $O(n \log n)$). The algorithm of Ma *et al* [15] can find the symbol periodicity with time tolerance window, but can not find the sequence periodicity and would have to combine the individual periodicity results for symbols to generate the approximate periodicity of the sequence where the periodic strength is given in (min-max) range. Since we run

the algorithm on the occurrence of the entire string (like 'TA' or 'AA'), we provide the exact periodic strength for the strength and not in some min-max range.

The third requirement, i.e., calculating the periodicity for a specific set of alternative strings, can be done by Elfeky's [2] and Ma's [15] algorithms, but we have to combine the results of the periodicity of individual symbols and join them by considering their starting positions in the time series; this would give us the periodic strength in (min-max) range. Hence, we may conclude that our algorithm is more flexible than any other periodicity detection approach to be applied to the biological data. The algorithm can be easily modified to work with the DNA sequences in addition to working for the regular time series data.

The periodicity detection in DNA sequence is of great significant in order to detect the nucleosome positioning. Many works have shown that there is dinucleotides periodicity in nucleosome–positioning sequences. Bina [3] demonstrated that there is periodic signal for AA/TT (10.26 bp) and GG/CC(10.bp) and AA dominates the occurrence of AA/TT. These results were obtained from the analysis of nucleosome sequences derived from simian virus 40 chromatin. Bina used statistical approach, which was based on the alignment of the DNA fragments with respect to their midpoints. Satchwell et al [4] detected a periodicity of 10bp in chicken nucleosome . They used fourier transformation to analyse the sequences. In another work, Herzel *et al* [5] reported that correlation functions of complete genomes revealed pronounced oscillations with period in the range of 10-11bp. Thastrom *et al* [6] discover periodicity in DNA sequences using ClUSTAL W for multiple sequence alignment. Segal *et al* [7] used probabilistic nucleosome-DNA model within statistical framework to compute the nucleosome organization intrinsic to the genomic DNA sequence. Interestingly, dinucleotide periodicity is detected in prokaryotes sequences, although they do not have nucleosomes [8]. Hosid [9] showed that sequence periodicity close to 11 is detected in *E.coli* . Only AA/TT dinucleotides contributed to overall dunucleotide periodicity in intergenic regions.

3 Periodicity Detection

Motivated by the above analysis, we developed a novel approach for periodicity detection by integrating suffix trees in the process. The proposed approach involves several phases as described next.

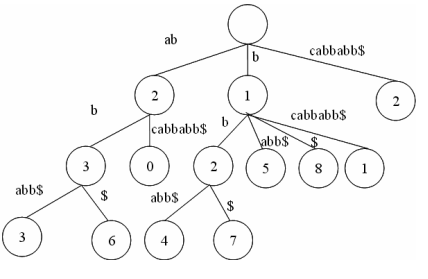


Fig. 1. The suffix tree for the string abcabbabb$

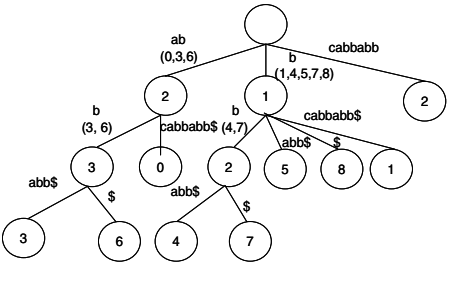


Fig. 2. The suffix tree for the string abcabbabb$ with substring occurrences

First Phase: Suffix Tree Based Representation: Suffix tree is a famous data structure [10] that has been proved to be very useful in string processing, e.g., [10, 11]. It can be efficiently used to find a substring in the original string, find the frequent substring; it can also be used to solve other substring matching problems. Each of the branches of the suffix tree represents a suffix of the original string. Hence, a suffix tree for a string of length 'n' has 'n' branches, and thus 'n' leaf nodes. For example, Figure 1 shows the suffix tree for the string 'abc abb abb$' where $ denotes the terminating symbol.

Each leaf node in the tree has an integer value showing the starting position of the substring achieved through the path from root to that leaf in the original string. Since there are exactly 'n' suffixes for a string, each starting at one of the index positions, there are 'n' leaf nodes in the tree. Each internal node (nodes that are neither leaf nor the root) has the integer value representing the length of the substring so far achieved while traversing from the root to the node. A suffix tree can have a maximum of 2n nodes, but mostly having periodicity and repetition in the time series, there are less than '2n' nodes in the suffix tree for these series.

We use the famous Ukonen algorithm [17] to construct the suffix tree for a given time series, which runs in linear time. The algorithm gives us the collection of 'edges', each having the starting node number, end node number, the first character index and the last character index and the value. For example, the edge from the root with label 'ab' in Figure 1 is represented as: starting node number: 0, end node number: 1, first character index: 0, last character index 1, and the value: 2. Thus, this edge can be represented in five-tuple (0, 1, 0, 1, 2). The subsequent edge labeling b is represented as (1, 4, 2, 2, 3), and the edge labeling cabbabb$ is represented as (1, 5, 2, 9, 0).

Second Phase: Periodicity Detection: Once we have the decorated suffix tree, we invoke the periodicity detection algorithm given in Figure 3. The performed process traverses the tree in bottom up fashion. During the traversal, each leaf node passes its value to the parent. The internal nodes after receiving the values from all of their children collect these in the collection called the *occurrence vector*. An occurrence vector is represented in our algorithm as 'occur_vect'. The tree in Figure 1 after performing this step is presented in Figure 2, where each internal node has its own occurrence vector. In fact, this vector shows the index positions in the original time series where this sequence appear. Since there are a maximum of 2n nodes in the suffix tree of a string of length 'n' and there are 'n' leaf nodes in the suffix trees, the tree should have significantly less than 'n' such vectors.

The second step is to calculate another vector for each of these occurrence vectors, which we call the difference vector (or 'diff_vect' in our algorithm). Let V be the occurrence vector of length m; $V = v_0, v_1, \ldots, v_{m-1}$

The difference vector 'D' would always have the length m-1 and would be

$$D = v1\text{-}v0, v2\text{-}v1, \ldots, vm\text{-}1\text{-}vm\text{-}2$$

Table 1. Occurrence and difference vectors for the sequence 'ab'

Index	0	1	2	3	4	5	6	7	8	9
occur_vect	0	3	12	16	21	24	27	38	45	48
diff_vect	3	9	4	5	3	3	11	7	3	

This is calculated by simply taking the difference of the consecutive values, and thus called the difference vector. The difference vectors contain the candidate periods. Each of these periods (with some exceptions mentioned in the sequel) is checked and the corresponding periodic strength is calculated.

Let vj and sj represent the jth entry in the difference and occurrence vectors, respectively, and 'i' be a positive integer. Then, we increment count(p, st) by 1 if and only if sk = sj + ivj, where i×vj ≤ max(occur_vect). The count(p, st) represents the frequency of the occurrence of a sequence starting from 'st' with a period value 'p'.

If the length of the time series is n, then the periodicity strength 'τ' is calculated as:

$$\tau(p, st) = \frac{count(p, st)}{\left\lfloor \frac{n-st}{p} + y \right\rfloor} \qquad y = \begin{cases} 1 & \text{iff } ((n\ st) \bmod p) > \text{edges value} \\ 0 & \text{otherwise} \end{cases}$$

The periodicity strength is the ratio between the frequency of a sequence's occurrences and the maximum possible number of occurrences for that sequence. For example, for the sequence abcabbabc$, τ(3, 2, 'c') is 2/3, as there are 2 occurrences of 'c', while the maximum possible occurrences are 3.

Adjustments to make the algorithm work with DNA Sequences: The already described algorithm [13] is not ready to deal with the DNA sequence data because of the special properties of the DNA sequences discussed earlier in Section 1. But the algorithm is flexible enough to work with such data with some minor changes. We are not looking for one sequence (or substring say TT) to be repeated at certain positions; however, we look for the occurrence of any of the alternative strings (AA/TT/TA). This is achieved by keeping a separate collection for alternative string occurrences and by applying the periodicity detection algorithm only on this combined collection.

Secondly, we adapted the window range of the periodic appearance of the string when analyzing the occurrence vector (as can be seen in line 2.9.1 of the CalculatePeriod algorithm in Figure 3). Hence, if the detected period is 10 then we also count the occurrence at a position which is in the range of 8-12bp from the occurrence of the pervious sequence (AA/TT/TA). Another problem is that not every occurrence of AA/TT/TA is to be periodic. Some of these instances is just randomly embedded in between two periodic instances. Our algorithm is able to deal with such problems. We deal with noise problem as follows, we measure the distance between the occurrence of any of AA/TT/TA instances and the following instance, if it is less than 8, then we ignore the second instance temporarily and check the distance with the instance which follows till we reach an instance in the range of 8-12 (as can be seen in line 2.3 of the CalculatePeriod algorithm in Figure 3).

The algorithm also does not consider periods greater than a certain value. It calculates the periodicity starting from each and every instance of the altenative string. This way, we do not miss any instance to be considered in the periodicity. Since we are looking for periodicity over the complete fragment, we ignore the period starting after a specific index position. The algorithm can also detect the periodicity strength, which allows us to include only the stonger periods (say those having periodic strength over 0.6).

To demonstrate the functionality of the algorithm, let us consider a nucleosomal DNA sequence of chicken as provided in [7]. This sequence contains 145 nucleotides

```
1. Initialize rootOccurSt and rootOccurLength and stack 's'
2. With each children edge 'e' (having stn = 0) of the root edge
   2.1. Sort children edges
   2.2. e.pntVal = 0              // parent value
   2.3. e.pntOccurSt = rootOccurSt
   2.4. e.pnOccurLength = rootOccurLength
   2.5. push e to stack 's'
3. while (stack is not empty)
   3.1. e = s.pop()
   3.2. if edge is already marked
      3.2.1. ProcessEdge(e)
      3.2.2. if e.pntOccurSt is blank
           3.2.2.1. e.pntOccurSt = e.occurSt
           3.2.2.2. e.pntOccurLength = e.occurLength
      3.2.3. else Join&Sort(e.pntOccurSt, e.pntOccurLength,e.occurSt, e.occurLength)
   3.3. else if edge has not been marked yet
      3.3.1. if e leads to leaf e.val = N-(e.lci-e.fci) + 1 + e.pntVal
           3.3.1.1 occur.add(e.val)
      3.3.2. else e.val = e.lci - e.fci + 1 + e.pntVal
           3.3.2.1. find and sort all children edges of e
           3.3.2.2. With each child edge 'ce'
              ce.pntVal = e.val
              ce.pntOccurSt = e.occurSt
              ce.pnOccurLength = e.occurLength
              s.push(ce)
      3.3.3. mark 'e'
4. Initialize an Edge ebio and set ebio.occurSt = bioOccurList,
      ebio.occurLength = bioOccurList.length,
     ebio.value = length of any alternative string
5. CalculatePeriod(ebio)
ProcessEdge: Edge e
1. Initialize chkStr = T.subString(e.occurSt, e.val)
2. if(chkStr matches any of alternatives)
   2.1. Add all e.occurLength number of occurences starting from
        e.occurSt to bioOccurList
CalculatePeriod: Edge e
1. current = e.occurSt
2. for ( i = 1; i < e.occurLength; i++)
   2.1. diffVal = current.next.val - current.val
   2.2. Initialize bioCurr = current.next
   2.3. while(diffValue < minPeriodValue AND bioCurr != null)
      2.3.1. diffValue = bioCurr.Value - current.value
      2.3.2. bioCurr = bioCurr.next
   2.4. if (diffVal < e.val OR diffVal > maxPeriodVal OR current.val > minStPos)
      2.4.1 current = current.next; continue from 2
   2.5. initialize p as candidate period
   2.6. p.val = diffVal, p.stPos = current.val, p.fci = p.stPos, p.len = e.val
   2.7. modRes = p.stPos mod p.val
   2.8. subCurrent = current, preSubCurValue = -5
   2.9. for ( int j = i; j<=e.occurLength; j++)
      2.9.1. if(modRes >= ((subCurrent.value mod p.periodValue) - relaxedRange)
           AND modRes <= ((subCurrent.value mod p.periodValue) + relaxedRange))
          2.9.1.1. if( (subCurrent.value - preSubCurValue) > 2*relaxedRange-1)
             2.9.1.1.1. p.freq++
             2.9.1.1.2. preSubCurValue = subCurrent.value
      2.9.2. subCurrent = subCurrent.next
   2.9. if ( (T.Len - 1 - p.stPos) mod p.val >= e.val) y = 1 else y = 0
   2.10. p.th = p.freq / Floor( (T.Len - 1 - p.stPos) / p.val + y)
   2.11. if (p.th >= minThreshold) add p to PeriodCollection
   2.12. current = current.next
```

Fig. 3. Algorithm for Periodicity Detection

(or it is 145 characters long). The sequence along with the periods found by the algorithm is presented in Figure 4. All the occurrences marked by either +, * or – are added to the combined occurrence collection which is like

```
Line 113

Period   StPos    Threshold        SymbolString     SymbolInMap
-------------------------------------------------------------
10       4        0.78             TT               *
12       16       0.82             TA               +
Both Period 10&12                                    #
Occurence of TT/TA/AA that is not periodic--
Symbol:          -**         **++          ++   **       ##  --
Sequence: TGCTTTGAGCACACAATAGAGGATCATGTTGAGTTCCTCATCAACCAATGC
Index:    0123456789012345678901234567890123456789012345678900
Symbol:           ##             --      **    ++      ##    **
Sequence: TCCAAGTCCGCCTCCATAGGGTTCTCCTTCAGCCATTCTCCTTCAGCTG
Index:    1234567890123456789012345678901234567890123456789
Symbol:      ++    **   -++**  --      ##           ---
Sequence: AACTGGAAGTGTTAAACATAGTGCCATTCAGAGTCTCTGAAAGCT
Index:    0123456789012345678901234567890123456789012345
```

Fig. 4. The chicken sequence and corresponding periods

Occur_vect(TT/AA/TA) = (3, 4, 14, 16, 28, 33, 42, 46, 54,…, 139, 140).

Now the difference between any two occurrence is a candidate period. The first period found by the algorithm is 10, starting at position 4. Its valid occurrences are

$$T(p, stPos) = T(10, 4) = 4, 14, 33, 42, 54, 72, 86, 92, 106, 114, 126$$
$$\tau(10,4) = 11 / \lfloor ((144 - 4) / 10) \rfloor = 11/14 = 0.78$$

Note that positions 33 and 42 are considered as valid because they occur inside ± 2 window of the exact positions 34 and 44, respectively.

4 Experimental Evaluation

For the experimental evaluation of the algorithm, we have used chicken and yeast data sets from [7]. These data sets contain 177 and 199 fragments of DNA, respectively, each is a nucleosome positioning sequence of around 150bp. In order to check the correctness of our algorithm, we used another dataset which contains random sequences from different chromosomes of chicken. We are concerned about the periodicity of AA/TT/TA in a dynamic window of 8 to 12 nucleotides.

The chicken and yeast data are known to have the periodicity of ~10 for AA/TT/TA. After applying the algorithm to the 177 chicken sequences, we got all possible periodicities for AA/TT/TA within each fragment. Then, we calculated the average of the periodicities for each fragment and the average for the whole set of fragments. For both chicken and yeast datasets, we were able to find the periodicity of AA/TT/TA in about 90% of the sequences, and the average periodicity is ~9.4 for chicken and ~9.3 for yeast. We also could apply the algorithm to the random dataset and only about 10% of the sequences showed periodicity. We believe this small percentage occurred by chance because some of the random sequences might be selected with nucleosome–positioning sequences. The percentage of each of the datasets showing the periodicity is depicted in Figure 5.

Many studies have shown that the periodicity of AA/TT in the eukaryotic is ~ 10. This periodicity has a lot of biological meaning in addition to its meaning in time

series analysis. In a "relaxed" double-helical segment of DNA, the two strands twist around the helical axis once every 10.55bp of the sequence. If a DNA segment under twist strain were to be closed into a circle by joining its two ends and then it is allowed to move freely, the circular DNA would contort into new shape, such as a simple figure-eight. Such a contortion is a supercoil. DNA supercoiling is important for DNA packaging within all cells. Because the length of DNA can be thousands of times that of a cell, packaging this genetic material into the cell or nucleus (in eukaryotes) is a difficult feat. Supercoiling of DNA reduces the space and allows for a lot more DNA to be packaged. Extra helical twists are positive and lead to positive supercoiling, while subtractive twisting causes negative supercoiling. According to Crick's formula for helicalDNA trajectories, periods above 10.55bp generate negatively supercoiled DNA, whereas lower periods induce positive supercoiling. Therefore, sequence periodicities reflect the characteristic superhelical density of genome DNA.

Our results confirm that periodicity of AA/TT/TA is around 10bp, which agrees with all the previous results reported by statistical approaches [3, 4, 7]. Positive supercoiling has shown to inhibit gene expression [12]. This result indicates that the nucleosomal DNA is not an active region. Since the hot regions which are transcribed continuously will be most of the time free from Histone proteins to let other proteins like RNA polymerase to access the DNA code. However, when the DNA is positively supercoiled, that means it will stay bounded to the histone proteins which means no transcription process will take place. We think that the periodicity of housekeeping genes for example will be greater than 10.55bp since it is active all the time.

We also argue that the periodicities detected in the chicken and yeast are not by chance because the algorithm did not discover any periodicity in the tested random dataset as expected. This means that our algorithm was able to deal with DNA sequences to identify periodicities when they do exist.

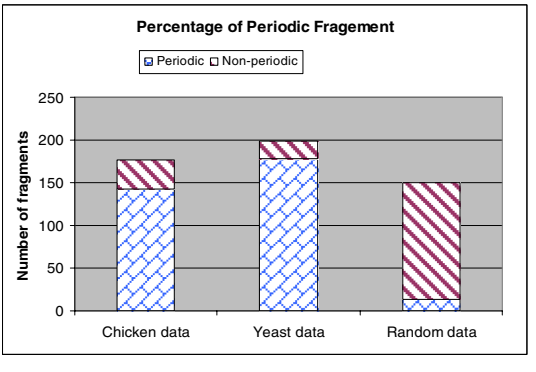


Fig. 5. Percentage of Periodic Fragment

5 Conclusions

DNA sequence analysis is getting more popular among biologists and statisticians. Most of the approaches to analyze DNA sequence are based on statistical techniques. In this work, we have applied an algorithm which is used in time series analysis to

detect periodicity in DNA sequence. We considered DNA sequence as time series data, and the occurrence of each nucleotide is an event at certain time. This way, we demonstrate that our approach is flexible enough to deal with DNA sequence which is more challenging than time series data. Currently, we are extending the algorithm to automatically work with both time series and DNA sequences. Also, we are planning to apply the algorithm on whole genome sequences.

References

1. Indyk, P., Koudas, N., Muthukrishan, S.: Identifying representative trends in massive time series data sets using sketches. In: Proc. of VLDB (2000)
2. Elfeky, M.G., Aref, W.G., Elmagramid, A.K.: Periodicity detection in time series databases. IEEE TKDE 17(7), 875–887 (2005)
3. Bina, M.: Periodicity of dinucleotide in Nucleosomes derived from simian virus 40 chromatin. Journal of Molecular Biology 235, 198–208 (1994)
4. Sarchwell, S.C., et al.: Sequence periodicities in checkin nucleosome core DNA. Journal of Molecular Biology 191, 659–675 (1986)
5. Herzel, H., Weiss, O., Trifonov, N.: 10-11 bp periodicities in complete genomes reflect protein structure and DNA folding. Bioinformatics 15(3), 187–193 (1999)
6. Thastrom, A., Bingham, L.M., Widom, J.: Nucleosomal locations of dominant DNA sequence motifs for histone-DNA interaction and nucleosome positioning. Journal of Molecular Biology 338, 695–709 (2004)
7. Segal, E., et al.: Agenomic code for nucleosome positioning. Nature, 442(17), 772–778
8. Herzel, H., Weiss, O., Trifonov, E.N.: Periodicity in complete genome of archaea suggests positive supercoiling. Journal of Biomol. Struct. Dyn. 16, 341–345 (1998)
9. Hosid, S., et al.: Sequence periodicity of Escherichia coli is concentrated in intergenic regions. BMC Molecular Biology 5(14) (2004)
10. Gusfield, D.: Algorithms on Strings, Trees, and Sequences. Cambridge Univ. Press, Cambridge (1997)
11. Grossi, R., Italiano, G.F.: Suffix trees and their applications in string algorithms. In: Proc. of South American Workshop on String Processing, pp. 57–76 (1993)
12. Gartenberg, M.R., Wang, J.C.: Positive supercoiling of DNA greatly diminishes mRNA synthesis in yeast. PNAS 89(23), 11461–11465 (1992)
13. Rasheed, F., Alhajj, R.: Using suffix trees for the periodicity detection in time series databases, Technical Report, Dept of Computer Science, University of Calgary (May 2007)
14. Elfeky, M.G., Aref, W.G., Elmagarmid, A.K.: WARP: Time Warping for Periodicity Detection. In: Proc. of IEEE ICDM, pp. 138–145 (2005)
15. Ma, S., Hellerstein, J.L.: Mining partially periodic event patterns with unknown periods. In: Proc. of IEEE ICDE, pp. 205–214 (2001)
16. Han, J., Yin, Y., Dong, G.: Efficient Mining of Partial Periodic Patterns in Time Series Database. In: Proc. of IEEE ICDE, p.106 (1999)
17. Ukkonen, E.: Online Construction of Suffix Trees. Algorithmica 4(3), 249–260 (1995)
18. Ahdesmäki, M., Lähdesmäki, H., Yli-Harja, O.: Robust Fisher's test for periodicity detection in noisy biological time series. In: Proc. of IEEE International Workshop on Genomic Signal Processing and Statistics, Tuusula, FINLAND (2007)

Analysis of Tiling Microarray Data by Learning Vector Quantization and Relevance Learning

Michael Biehl[1], Rainer Breitling[2], and Yang Li[2]

[1] Institute of Mathematics and Computing Science, University of Groningen, P.O. Box 800, 9700 AV Groningen, The Netherlands
m.biehl@rug.nl
[2] Groningen Bioinformatics Centre, University of Groningen, Kerklaan 30, 9751 NN Haren, The Netherlands

Abstract. We apply learning vector quantization to the analysis of tiling microarray data. As an example we consider the classification of *C. elegans* genomic probes as intronic or exonic. Training is based on the current annotation of the genome. Relevance learning techniques are used to weight and select features according to their importance for the classification. Among other findings, the analysis suggests that correlations between the perfect match intensity of a particular probe and its neighbors are highly relevant for successful exon identification.

1 Introduction

Tiling microarrays are used to interrogate genome-wide transcriptional activity at high resolution in an unbiased fashion. This technology is rapidly becoming one of the most important high-throughput functional genomic assays [1]. One important application is the comprehensive detection of transcribed regions in the genome, which has changed our view of the gene expression landscape and lead to the detection of many new genes [2]. At regular intervals along the genome, one places probes that measure the expression level at this position. The main goal of interpreting tiling data is to discriminate outlier probes (corresponding to expressed regions) from the predominant background or noise signals. This is complicated by the fact that the majority of transcribed sequences are present at levels just above the background [3]. Moreover, background signal intensity is strongly probe-specific. Different statistical algorithms have been applied for detecting transcribed regions in tiling array data. For example, a robust pseudo-median estimator together with heuristic maxgap and minrun parameters [4] was used for an in-depth analysis of human chromosome 21 and 22 tiling data. Bertone et al. [5] employed binomial theory using a p-value cut-off with maxgap/minrun for human whole-genome tiling data. A moving-window robust principal component analysis (rPCA) with Mahalanobis distance was used by Schadt et al. [1] for a tiling microarray experiment with multiple human samples. More recently, hidden Markov model approaches were also applied to this problem, see e.g. [6]. For the purposes of the present paper, we consider the task

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 880–889, 2007.

of detecting transcribed regions as a classification problem, aiming at discriminating transcribed and non-transcribed probes along the genome. The partially validated knowledge about array data such as gene annotation is used to assist the analysis of genomic tiling data in a supervised way.

We implement the classification by means of learning vector quantization [7], a particularly intuitive and flexible tool which has been applied in a variety of areas [8]. One of its most attractive features is the possibility to incorporate adaptive metrics into the training procedure. So-called relevance learning schemes [9,10,11,12,13] employ a similarity measure in which features are weighted according to their importance for the classification. Results provide insights into the nature of the problem and allow for immediate interpretation of the classifier.

2 The Classification Problem

Our example dataset contains expression measurements from multiple *C. elegans* samples hybridized to the Affymetrix 1.0R tiling array. Probes of 25 base pairs are tiled end-to-end along the entire genome, resulting in a total of 3 million data points per sample. In addition to probes that correspond to the genome sequence (perfect match probes, PM), the array also contains so-called mismatch probes (MM), which sometimes are suggested to help estimating the background signal at a particular genome position. All probes were matched to the most recent version of the *C. elegans* genome and labeled as either exonic (if they correspond to an annotated exon region of the genome) or intronic (if they correspond to an intron or intergenic region). This labeling is not error-free, because some genes are transcriptionally silent (their exons are not expressed), and new genes are regularly discovered (resulting in intergenic regions being expressed). In Sec. 5 we will discuss the effect of these two sources of mislabeling.

2.1 Features for Classification

We randomly pick a genome region $[4413428 : 4540601]$ in chromosome 3 of *C.elegans*. It contains 4120 probes, with 2587 and 1533 probes corresponding to exonic and intronic/intragenic regions, respectively. We consider the following features for each probe μ: The median signal of the perfect match probe across all samples (PM_μ), the corresponding mismatch signal (MM_μ), the Pearson correlation between a probe and its left and right neighbors ($CC.PM_{\mu,\mu-1}, CC.PM_{\mu,\mu+1}$), the calculated melting temperature (Tm_μ) according to the method described in reference [14]. Furthermore, because transcripts usually span larger areas of the genome, the intensities of neighboring probes could also be informative for detecting transcribed regions. We take this into account by adding the PM and MM values of the neighboring ± 2 probes to the feature set ($PM_{\mu-2}, PM_{\mu-1}, \ldots$). Finally, for each probe we tested if it shows significant strain or stage effect using ANOVA analysis. The resulting $-\log(p)$-value was used as a feature that indicates if a probe shows biological variation, the reasoning being that only expressed probes should have significant strain and stage effects, while noise should

be randomly distributed. All of these features are biologically motivated and can individually discriminate between expressed and non-expressed probes to some extent, but our results will show that not all of them are equally informative.

The above mentioned features will be referred to in the following order:

(1) $PM_{\mu-2}$, (2) $PM_{\mu-1}$, (3) PM_{μ}, (4) $PM_{\mu+1}$, (5) $PM_{\mu+2}$,
(6) $MM_{\mu-2}$, (7) $MM_{\mu-1}$, (8) MM_{μ}, (9) $MM_{\mu+1}$, (10) $MM_{\mu+2}$,
(11) $CC.PM_{\mu,\mu-2}$, (12) $CC.PM_{\mu,\mu-1}$, (13) $CC.PM_{\mu,\mu+1}$, (14) $CC.PM_{\mu,\mu+2}$
(15) $CC.MM_{\mu,\mu-2}$, (16) $CC.MM_{\mu,\mu-1}$, (17) $CC.MM_{\mu,\mu+1}$, (18) $CC.MM_{\mu,\mu+2}$
(19) $Tm_{\mu-2}$, (20) $Tm_{\mu-1}$, (21) Tm_{μ}, (22) $Tm_{\mu+1}$, (23) $Tm_{\mu+2}$, (24) $-\log(p)$.

2.2 Data Set and Validation Procedure

In total, a set of $M = 4120$ examples, i.e. labeled probes, is considered, which we denote as $\mathbb{D} = \{\boldsymbol{\xi}^{\mu}, S_T^{\mu}\}_{\mu=1}^{M}$. Here, the annotated class membership of probe μ is denoted as $S_T^{\mu} = 0$ (intron) or $S_T^{\mu} = 1$ (exon), respectively. Components of the vectors $\boldsymbol{\xi}^{\mu} \in \mathbb{R}^N$ $(N = 24)$ are obtained from the above listed features by means of a z-transformation. The transformed values display zero mean and unit variance over the set of available data, i.e. $\sum_{\mu} \xi_i^{\mu} \,/\, M = 0$ and $\sum_{\mu} (\xi_i^{\mu})^2 \,/\, M = 1$. The transformation facilitates a straightforward interpretation of the relevance factors which we define and consider in Sec. 4.1.

We consider the construction or training of classifiers from $P = 3000$ randomly selected examples while the remaining 1120 data serve as a test set. By comparing the classifier output and the annotated labels S_T^{μ} we determine the fraction ε_{train} of misclassified examples in the training set. Analogously, ε_{test} quantifies the over-all error rate in the test set. In addition, we will consider the class specific training errors $\varepsilon_{train}^{(0)}, \varepsilon_{train}^{(1)}$ and the test errors $\varepsilon_{test}^{(0)}, \varepsilon_{test}^{(1)}$ with respect to only class 0 (intron) or class 1 (exon) data, respectively. All results given here are obtained on average over 50 random splits of $\mathbb{D}$ into training and test set. The additional average reduces the influence of lucky set compositions.

3 Fixed Metrics Classifiers

Many classifying systems are based on a distance measure which quantifies the similarity of a given feature vector with representatives of the classes. We will first consider the use of a fixed measure which corresponds to the standard L_1 metric. For two arbitrary vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^N$ we define

$$d(\boldsymbol{x}, \boldsymbol{y}) \;=\; \sum_{j=1}^{N} |x_j - y_j| \quad . \tag{1}$$

For all considered classifiers we have observed that the use of this so-called Manhattan distance yields slightly better performance on our data set than the quadratic Euclidean distance measure. For generalized L_q metrics with $q \geq 3$ the performance further deteriorates.

First we consider prototype based schemes which use (1) as an a priori defined, fixed measure of similarity. For comparison we have also studied the standard k-nearest neighbor (KNN) classification scheme [15]. Corresponding leave-one-out

estimates of the test error are given in Table 1 for the cases $k = 1$ and $k = 13$ which turns out to yield the best results. We furthermore obtained preliminary results for the support vector machine, i.e. a large margin linear classifier. Its performance ($\varepsilon_{test} \approx 11\%, \varepsilon_{test}^{(0)} \approx 5\%, \varepsilon_{test}^{(1)} \approx 21\%$) is comparable to that of the best KNN system.

Table 1. **a)** Leave-one-out error estimates of the KNN classifier. **b)** Training and test error estimates for the CCM classification scheme. All errors are given in %.

a) KNN	ε_{test}	$\varepsilon_{test}^{(0)}$	$\varepsilon_{test}^{(1)}$
k=1	15.6	11.7	22.2
k=13	10.6	3.0	23.5

b) CCM	ε	$\varepsilon^{(0)}$	$\varepsilon^{(1)}$
training set	12.5	4.9	25.3
test set	12.5	5.0	25.3

3.1 Class Conditional Means

The KNN approach requires the explicit storage of a large set of examples and involves the evaluation of many distances for each classification event. Hence, it is preferential to represent the data set by only a few prototype vectors which capture essential properties of the classes. Novel data can then be labeled according to a computationally cheaper nearest prototype classification (NPC) scheme.

The simplest set of prototypes obtained from P examples is given by the class conditional mean (CCM) in each class, i.e. $\boldsymbol{m}^{(S)} = \sum_{\mu=1}^{P} \boldsymbol{\xi}^{\mu}\, \delta(S_T^{\mu}, S) \,/ P_S$ for $S = 0, 1$. Here, $\delta(k, l) = 1$ if $k = l$ and 0 else, and the number of training examples from class S is denoted as $P_S = \sum_{\mu} \delta(S_T^{\mu}, S)$. The resulting classifier defines a linear decision boundary and assigns a vector $\boldsymbol{\xi}$ to class 1 if $d(\boldsymbol{m}^{(1)}, \boldsymbol{\xi}) \leq d(\boldsymbol{m}^{(0)}, \boldsymbol{\xi})$ and to class 0 else. While individual samples show a large variability, we observe that the CCM vectors of class 1 (class 0) consist of only positive (negative) components. Table 1 shows that the CCM system outperforms the KNN classifier for $k = 1$ in terms of the over-all test error.

3.2 Learning Vector Quantization

Beyond the use of CCM prototypes, we apply learning vector quantization (LVQ) for the identification of class representatives. LVQ was originally proposed by Kohonen [7] and has been used in a variety of problems due to its flexibility and conceptual clarity, see [8] for up-to-date references. We first resort to the original LVQ1 [7] which will be extended by heuristic relevance learning in Sec. 4.1.

A set of vectors $\left\{\boldsymbol{w}^1, \boldsymbol{w}^2, \ldots, \boldsymbol{w}^k\right\}$ with $\boldsymbol{w}^j \in \mathbb{R}^N$ is used to parameterize an NPC scheme. The prototypes represent classes according to the associated labels $S^j \in \{0, 1\}$. We will denote the number of vectors $\boldsymbol{w}^j$ assigned to classes 0 and 1 by k_o and k_1, respectively. This assignment as well as the total number of prototypes $k = k_o + k_1$ are specified prior to learning.

At each time step t of an iterative training procedure, one example $\{\boldsymbol{\xi}^{\mu}, S_T^{\mu}\}$ is selected randomly from the training set ($1 \leq \mu \leq P$). Its distances $d(j, \mu) = d(\boldsymbol{\xi}^{\mu}, \boldsymbol{w}^j(t))$ from all current vectors $\boldsymbol{w}^j(t)$ are evaluated and we identify the

closest of all prototypes. In LVQ1, only this so-called winner $\boldsymbol{w}^J(t)$ with $d(J,\mu) = \min_k \{d(k,\mu)\}$ is updated according to

$$\boldsymbol{w}^J(t) = \boldsymbol{w}^J(t) + \eta_w\, \psi(S_T^\mu, S^J)\, \left(\boldsymbol{\xi}^\mu - \boldsymbol{w}^J(t)\right) \text{ with } \psi(s,t) = \begin{cases} +1 \text{ if } s = t \\ -1 \quad \text{else.} \end{cases} \quad (2)$$

The update is towards (away from) the actual input $\boldsymbol{\xi}^\mu$ if the class labels of winner and example agree (disagree). Initially, we place prototypes close to the origin with a small random offset.

The learning rate η_w controls the step size of the iteration. Numerical results given in the following correspond to the choice $\eta_w = 10^{-2}$. Note that our main findings display only a weak dependence on rates in the range $10^{-4} \leq \eta_w \leq 10^{-2}$. The potential further improvement of the performance by suitable time dependent learning rates will be addressed elsewhere.

In the simplest setting, one prototype is employed per class, i.e. $k_o = k_1 = 1$. After about $t/P = 10$ randomized sweeps through the data the system has converged. It exhibits slightly larger training and test errors than the simple CCM classifier. The heuristic LVQ1 does not directly aim at minimizing the classification error and, hence, it is not guaranteed to improve the performance over the simple CCM system. However, the complexity and power of the LVQ system can be increased by introducing more prototypes. Figure 1 (left) shows example learning curves of different configurations. Averaged over-all test errors are displayed as a function of training time. The example choice $k_o = 1, k_1 = 2$ yields no significant improvement, while the system with $k_o = k_1 = 3$ outperforms the CCM. LVQ1 with $k_o = k_1 = 6$ yields a performance which is comparable with the best KNN system, however at much lower computational cost.

Table 2 summarizes the performance in several example settings. Note that the larger variability of class 1 (exon) data is reflected in the observation that $\varepsilon_{test}^{(1)} > \varepsilon_{test}^{(0)}$, in general. Consequently, configurations with $k_1 > k_o$ are to be preferred over systems that assign more prototypes to class 0. This observation agrees with recent theoretical findings within a model situation [16].

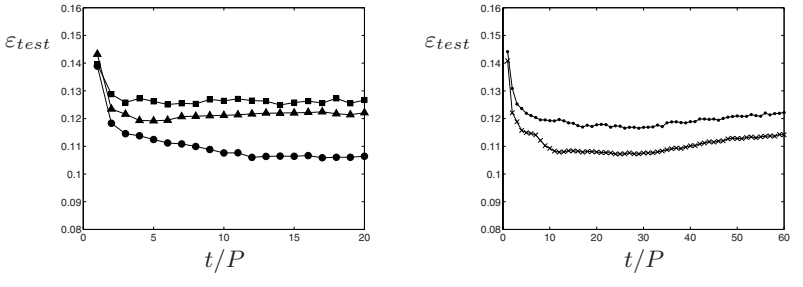


Fig. 1. Averaged test error as a function of the number t/P of randomized sweeps through the training set. **Left:** LVQ1 training with $k_o = 1, k_1 = 2$ (squares), $k_o = k_1 = 3$ (triangles), and $k_o = k_1 = 6$ (circles). **Right:** RLVQ training with local relevances and $k_o = 1$ and $k_1 = 2$ (upper) and with global relevances for $k_o = k_1 = 6$ (lower curve).

Table 2. Test error estimates (in %) of LVQ1 systems without relevance learning. Training errors are typically on the order 0.1% smaller than the test errors.

k_o	k_1	ε_{test}	$\varepsilon_{test}^{(0)}$	$\varepsilon_{test}^{(1)}$
1	1	12.9	2.5	30.6
3	3	12.1	3.4	26.7
6	6	10.7	4.3	21.5

k_o	k_1	ε_{test}	$\varepsilon_{test}^{(0)}$	$\varepsilon_{test}^{(1)}$
1	2	12.7	5.6	24.6
2	1	13.4	2.0	32.8

Although we do not observe over-fitting in the considered systems, one cannot expect the performance to improve further with even larger k_o, k_1. In fact, for very large k, the behavior of the nearest neighbor classifier should be recovered.

4 Adaptive Metrics Classifiers

The a priori choice of an appropriate distance measure is crucial for the success of LVQ and similar systems. In a particularly elegant and successful framework the metric is adapted in the course of training: Relevance learning vector quantization schemes update the prototypes and, at the same time, search for a discriminative similarity measure.

Here we follow a standard approach which was suggested and put forward in [9,10]. It modifies the distances (1) by attaching a scaling or relevance factor to each dimension in feature space, see Sec. 4.1. The term global relevances will be used when a unique set of factors is assigned to all prototypes. In this case, the decision boundaries of the LVQ classifier remain piecewise linear. The extension to local relevances with an independent set of factors for each prototype is formally straightforward. However, the resulting classification boundaries of the NPC scheme become curved, i.e. piecewise quadratic. Cases of intermediate complexity, e.g. with class-wise relevances, are straightforward to introduce but will not be considered here. The adaptation of global relevances was first suggested in [9]. Local relevances have been studied and applied in, e.g., [11,12,13].

After training, the resulting relevances implement a weighting scheme which allows to read off the importance of features for the classification. If, for instance, the factor attached to dimension j in feature space becomes *zero*, the corresponding feature might as well be omitted from the data set. Thus, relevance learning can serve as a tool for the detection of, e.g., noisy features which are of little use or can even deteriorate the classification performance if included.

In the following, we discuss two example scenarios only: global relevances in a setting with six prototypes per class and local relevance learning with $k_o = 1$ and $k_1 = 2$. We focus on the insights that relevance learning provides into the classification problem. A more detailed comparison of local, global, and class-wise relevance training will be given elsewhere, including the optimization of performance by choice of k_o, k_1, time dependent learning rate etc.

4.1 Relevance Learning Vector Quantization

We consider a generalized Manhattan distance of the form

$$d^i_\lambda(\boldsymbol{w}^i, \boldsymbol{\xi}) = \sum_{j=1}^N \lambda^i_j \left| w^i_j - \xi_j \right| \quad , \tag{3}$$

where the adaptive relevance factors λ^i_j are restricted to non-negative values and obey the normalization $\sum_{j=1}^N \lambda^i_j = 1$. The special case $\lambda^i_j = 1/N$ for all $j = 1, \ldots N$ is analogous to the original L_1-measure.

Our heuristic realization of relevance learning vector quantization (RLVQ) follows closely the prescription of [9], where it is exemplified in terms of the squared Euclidean distance. In parallel with the LVQ1 update (2) for the winning prototype $\boldsymbol{w}^J$, its relevance factors are adapted as follows:

$$\tilde{\lambda}^J_j(t) = \lambda^J_j(t\text{–}1) - \eta_\lambda \psi(S^\mu_T, S^J) \left| \xi^\mu_j - w^J_j(t) \right| ; \; \lambda^J_j(t) = \frac{\max\{0, \tilde{\lambda}^J_j(t)\}}{\sum_{k=1}^N \max\{0, \tilde{\lambda}^J_k(t)\}}, \tag{4}$$

where the second step implements the non-negativity condition and the required normalization. In the case of global relevances, all $\lambda^i_j(t)$ have to be set equal to $\lambda^J_j(t)$ after performing (4), in addition.

The prescription decreases relevance factor λ^J_j if, for instance, the winning prototype $\boldsymbol{w}^J$ does represent the correct class but the contribution $\left| \xi^\mu_j - w^J_j \right|$ to $d_{\lambda^J}(\boldsymbol{w}^J, \boldsymbol{\xi}^\mu)$ is relatively large. On the contrary, the weight of a feature with relatively small $\left| \xi^\mu_j - w^J_j \right|$ is increased in such a case. Thus, the measured distance will be smaller when presenting the same or a similar feature vector in the future and the probability for correct classification increases.

The learning rate η_λ controls the magnitude of relevance updates. Empirically, it has proven advantageous to set $\eta_\lambda \ll \eta_w$ in comparison with the step size of prototype updates. Numerical results presented here correspond to the choice $\eta_w = 10^{-2}, \eta_\lambda = 10^{-5}$. As in LVQ1 we initialize prototypes randomly close to the origin. Prior to learning, all relevances are set to $1/N$.

Figure 1 (right) displays the evolution of the over-all test error with the number of randomized sweeps through the data set. Initially, errors decrease in the course of learning, as prototypes and relevances adapt to the examples. Test and training errors reach a common minimum after a number of sweeps through the training set. Table 3 specifies the corresponding minimal test errors.

The learning curve for the system with six prototypes per class is shown in Fig. 1 (right), relevance profiles are displayed in Fig. 2. Its performance in the minimum of the learning curve is practically identical with that of the same system without relevances, cf. Table 2. Note, however, that relevance learning has reduced the number of features by effectively disregarding features 15–23, i.e. the correlations of neighboring mismatch intensities and all melting temperatures. If further training is performed, the relevance profile becomes more pronounced and RLVQ over-simplifies the classifier, see Fig. 2 (right panel). As a consequence, training and test errors mildly increase. In our example, the system saturates at $\varepsilon_{test} \approx 11.5\%$. This performance is achieved by using only features $2, 3, 4$ $(PM_\mu, PM_{\mu\pm1})$ and 8, the mismatch probe intensity MM_μ.

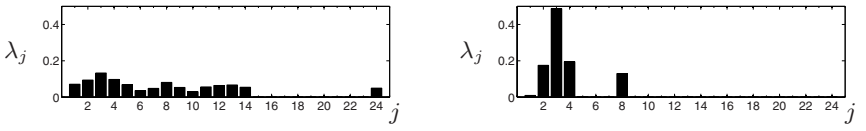


Fig. 2. Global relevance profiles in RLVQ with $k_0 = k_1 = 6$. **Left:** Relevances corresponding to the minimum of the learning curve. **Right:** Over-simplified relevances as observed after 60 sweeps.

The non-monotonic learning behavior suggests to introduce regularization terms into the update rules, which control the uniformity of the relevance profile. Here, we resort to the simpler early stopping strategy in order to obtain the best achievable performance. The effect of over-simplification is also observed in training with local relevances which we discuss in terms of the example case $k_o = 1, k_1 = 2$. The optimal performance of local RLVQ is superior compared with that of original LVQ1 in the same setting, cf. Table 2 and Fig. 1 (left panel). Thus, the introduction of relevances increases the complexity and improves the performance of the classifier. The local relevance profiles in the minimum of the learning curve are shown in the left panel of Fig. 3. Note that the resulting distance measures used for the identification of the two classes differ significantly. For instance, features 11–23 (all correlations and melting temperatures) are effectively disregarded by the class 0 prototype, while the class 1 prototypes assign relatively large relevances to perfect match intensity correlations (11–14).

Table 3. Test errors in the minima of learning curves for two different RVLQ scenarios

relevances	k_o	k_1	ε_{test}	$\varepsilon_{test}^{(0)}$	$\varepsilon_{test}^{(1)}$
local	1	2	11.8	4.5	24.0
global	6	6	10.7	3.9	22.6

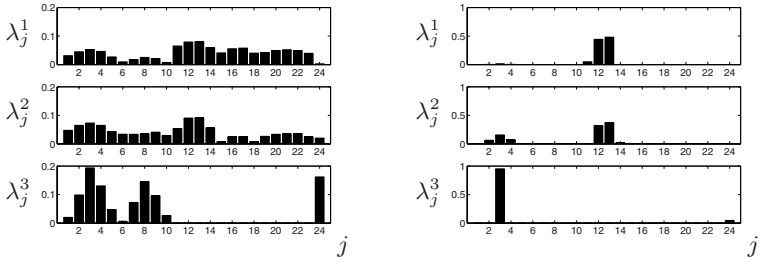


Fig. 3. Results of local RLVQ with $k_0 = 1$ and $k_1 = 2$. **Left:** Relevance factors in the minimum of the learning curve; the top two profiles correspond to class 1 prototypes, the bottom one to class 0. **Right:** Same as left panel, but after 140 training sweeps.

Again, the relevance profiles become more pronounced and RLVQ over-simplifies the classifier in later stages of the training process, see Fig. 3 (right panel). As a consequence, training and test errors increase. In the example, the over-all test error saturates at $\varepsilon_{test} \approx 12.6\%$, a value which is still comparable with that of the CCM result. However, the over-simplified RLVQ classifier achieves this performance by using mainly three components of the data: $j = 3(PM_\mu), 12(CC.PM_{\mu-1})$, and $13(CC.PM_{\mu+1})$. We observe that, indeed, precisely these features are selected when applying larger learning rates η_λ.

5 Discussion and Outlook

Our results demonstrate the usefulness of RLVQ as a tool for tiling microarray data analysis. It is very interesting to observe how the unbiased, data driven RLVQ procedure assigns the highest relevance to those features that are also biologically expected to be the most informative. In addition to the obvious informative feature PM_μ, features like $MM_\mu, CC.PM_{\mu,\mu-1}$, and $CC.PM_{\mu,\mu+1}$ are also selected. The latter two are of particular importance in the identification of exons. The large difference in test error rate for the two classes also has a biological basis. It is due to the mislabeling problem discussed in Sec. 2. It is relatively unlikely that new genes are discovered, so the intergenic regions (class 0) are mostly labeled correctly. On the other hand, only about 50-80% of genes are expressed at detectable levels at any given time, while the rest are transcriptionally silent. Thus, between 20-50% of class 1 probes are expected to be mislabeled and the apparent prediction error will be higher for class 1.

In forthcoming projects we will address, among other extensions, RLVQ schemes which are capable of taking into account correlations between different features by means of relevance matrices [12,13]. The aim is to further improve the classification performance and to obtain novel insights into the characteristics of exon and intron probes. The investigation of false introns should be of particular interest with respect to the potential detection of new genes. In such an analysis, the confidence of the classification should be taken into account, which, in LVQ, is straightforward to quantify in terms of distances.

Being computationally cheap, RLVQ can be easily applied to whole-genome tiling data (with millions of probes) while this is very challenging for other methods like the SVM. Furthermore, the small number of tunable parameters makes it easy to apply RLVQ to a broad range of organisms and technological platforms.

References

1. Schadt, E., Edwards, S., Guha Thakurta, D., Holder, D., Ying, L., Svetnik, V., Leonardson, A., Hart, K., Russel, A., Li, G., et al.: A comprehensive transcript index of the human genome generated using microarrays and computational approaches. Genome Biol. 5, R73 (2004)

2. Johnson, J., Edwards, S., Shoemaker, D., Schadt, E.: Dark matter in the genome: evidence of widespread transcription detected by microarray tiling experiments. Trends Genet. 21, 93–102 (2005)
3. Royce, T., Rozowsky, J., Bertone, P., Samanta, M., Stolc, V., Weissman, S., Snyder, M., Gerstein, M.: Issues in the analysis of oligonucleotide tiling microarrays for transcript mapping. Trends Genet. 21, 466–475 (2005)
4. Kampa, D., Cheng, J., Kapranov, P., Yamanaka, M., Brubaker, S., Cawley, S., Drenkow, J., Piccolboni, A., Bekranov, S., Helt, G., et al.: Novel RNAs identified from an in-depth analysis of the transcriptome of human chromosomes 21 and 22. Genome Res. 14, 331–342 (2004)
5. Bertone, P., Stolc, V., Royce, T., Rozowsky, J., Urban, A., Zhu, X., Rinn, J., Tongprasit, W., Samanta, M., Weissmann, S., et al.: Global identification of human transcribed sequences with genome tiling arrays. Science 306, 2242–2246 (2004)
6. Du, J., Rozowsky, J., Korbel, J., Zhang, Z., Royce, T., Schulz, M., Snyder, M., Gerstein, M.: A supervised hidden markov model framework for efficiently segmenting tiling array data in transcriptional and chIP-chip experiments: systematically incorporating validated biological knowledge. Bioinformatics 22, 3016–3024 (2006)
7. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (1997)
8. Neural Networks Research Centre, Helsinki: Bibliography on SOM and LVQ, Helsinki University of Technology (2002), On-line: http://liinwww.ira.uka.de/bibliography/Neural/SOM.LVQ.html
9. Bojer, T., Hammer, B., Schunk, D., von Toschanowitz, K.T.: Relevance determination in learning vector quantization. In: Verleysen, M. (ed.) Europ. Symp. on Artificial Neural Networks 2001, pp. 271–276. d-facto publications (2001)
10. Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Networks 15, 1059–1068 (2002)
11. Hammer, B., Schleif, F.M., Villmann, T.: On the generalization ability of prototype based classifiers with local relevance determination. Technical Report, Clausthal University of Technology Ifi-05-14 (2005)
12. Schneider, P., Biehl, M., Hammer, B.: Relevance matrices in LVQ. In: Verleysen, M. (ed.) Europ. Symp. on Artificial Neural Networks 2007, d-side, pp. 37–42 (2007)
13. Schneider, P., Biehl, M., Schleif, F.M., Hammer, B.: Advanced metric adaptation in generalized LVQ for classification of mass spectrometry data. In: Workshop on the Self-Organizing-Map, WSOM 2007, Univ. Bielefeld (in press, 2007)
14. SantaLucia Jr., J.: A unified view of polymer, dumbbell, and oligonucleotide DNA nearest neighbor thermodynamics. Proc. Natl. Acad. Sci. USA 95, 1460 (1998)
15. Duda, R., Hart, P., Stork, D.: Pattern Classification. Wiley, New York (2000)
16. Witoelar, A., Biehl, M., Hammer, B.: Learning vector quantization: generalization ability and dynamics of competing prototypes. In: Workshop on the Self-Organizing-Map, WSOM 2007, Univ. Bielefeld (in press, 2007)

Discriminating Microbial Species Using Protein Sequence Properties and Machine Learning

Ali Al-Shahib[1,2], David Gilbert[2], and Rainer Breitling[3]

[1] Biomedical Informatics Signals and Systems Research Laboratory, Department of Electronic, Electrical and Computer Engineering, The University of Birmingham, Birmingham, UK
a.alshahib@bham.ac.uk
[2] Bioinformatics Research Centre, Department of Computing Science, University of Glasgow, Glasgow, UK
[3] Groningen Bioinformatics Centre, Groningen Biomolecular Sciences and Biotechnology Institute, University of Groningen, Haren, The Netherlands

Abstract. Much work has been done to identify species-specific proteins in sequenced genomes and hence to determine their function. We assumed that such proteins have specific physico-chemical properties that will discriminate them from proteins in other species. In this paper, we examine the validity of this assumption by comparing proteins and their properties from different bacterial species using Support Vector Machines (SVM). We show that by training on selected protein sequence properties, SVMs can successfully discriminate between proteins of different species. This finding takes us a step closer to inferring the functional characteristics of these proteins.

1 Introduction

Species divergence is mainly caused by variation in gene and protein sequences but also by differences in the set of genes that is present in a particular species. Proteins that are specific for a particular species may be responsible for its adapted phenotype, e.g. its ability to act as a pathogen or its resistance to a certain drug. Identifying species-specific proteins is thus a relevant aim, and here we make a small contribution towards its achievement.

In this paper, we have compared the proteins of seven different bacterial species by extracting numerous protein sequence properties using state-of-the-art Support Vector Machines. To our surprise, we find that proteins of different species are significantly dissimilar and can be distinguished based on sequence properties selected prior to classification. This discrimination does not rely on any homology criteria but is based only on the biophysical characteristics encoded in the sequence. We have also constructed a phylogenetic tree based on the results of the comparisons, and compared it to the well-documented 16S rRNA dendrogram of the same bacteria. Interestingly, there is no detectable similarity between the two dendrograms.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 890–897, 2007.

2 Methodology

Seven Sexually Transmitted Disease-causing bacteria were used for this study. The protein sequences were obtained from the Los Alamos National Laboratory [4]. Table 1 shows the total number of proteins of each species in our bacterial database.

Table 1. Total number of proteins of each species in our database

Name	Total number of proteins
Chlamydia trachomatis (CT)	902
Haemophilus ducreyi (HD)	1830
Mycoplasma genitalium (MG)	485
Neisseria gonorrhoeae (NG)	2188
Streptococcus agalactiae (SAG)	2177
Treponema pallidum (TP)	1051
Ureaplasma urealyticum (UU)	614

All the functionally known proteins of every genome and their sequences were collected and 2579 sequence properties for every protein were extracted. These include some global properties (e.g. isoelectric point and molecular weight), the frequency and total number of each amino acid, the frequency and total number of certain sets of amino acids (e.g. hydrophobic, charged, polar), the number and size of continuous stretches of each amino acid or amino acid set, secondary structure predictions obtained using the Prof algorithm [5], the position of putative transmembrane helices predicted using TMHMM [6], and that of disordered regions obtained using DisEMBL [7]. A full list of the properties is available at `http://www.dcs.gla.ac.uk/~alshahib/features.pdf`.

Normalization of the features was performed for all features of all proteins at once, instead of normalization for every individual genome. This is important as performing normalization of the features for every individual species will cause slight differences in the scales of the features when combined with the normalized feature protein values of any other species. These slight differences will cause unjustified discrimination of the species. Thus it was appropriate to rescale each feature by its mean and variance [15] with respect to all proteins from all species.

Proteins of every species were combined with every other species (pairwise). A total of 21 pairwise species comparisons were performed. For every comparison, undersampling of the negative examples to equal the positive examples [8] was performed. Five training and test sets were then generated for every comparison and each was homology-corrected as described in [10]. This is vital because once the training and test sets are divided for training and testing, one must make sure that similarity between proteins on both sets is minimal. In other words, predictions must not be made based on homology of proteins in the training and test sets but rather on non-homologous proteins. We have thus implemented

a recursive BLAST strategy to assign proteins that show significant sequence similarity to each other to the same set (either test or training). For details see [10].

For every training set of every pairwise species comparison, feature selection was performed using our FrankSum method [9]. Finally, Support Vector Machine classification was performed for every pairwise species comparison and the AUCs were recorded. A polynomial kernel of order 3 with a C value of 1 was used for SVM classification. The WEKA machine learning package was used for this task [11].

3 Results

For every pairwise species comparison, five AUCs (obtained on 5 test set–training set combinations) were recorded. Figure 1 shows the performances achieved when comparing proteins from different source species.

All species pairs can be readily discriminated, except *H. ducreyi* and *S. agalactiae*, and *H. ducreyi* and *M. genitalium*. The median discrimination performance is as high as 91% for the proteins of *Treponema pallidum* and *Ureaplasma urealyticum* compared against all other species. The worst performance, for *Haemophilus ducreyi*, is still a surprising 83%. For any randomly selected pair of proteins from two species, a correct assignment to its species of origin will be possible in 85% of cases. This is achieved based solely on the sequence properties described above. It is all the more unexpected, as the bacteria that we analyse

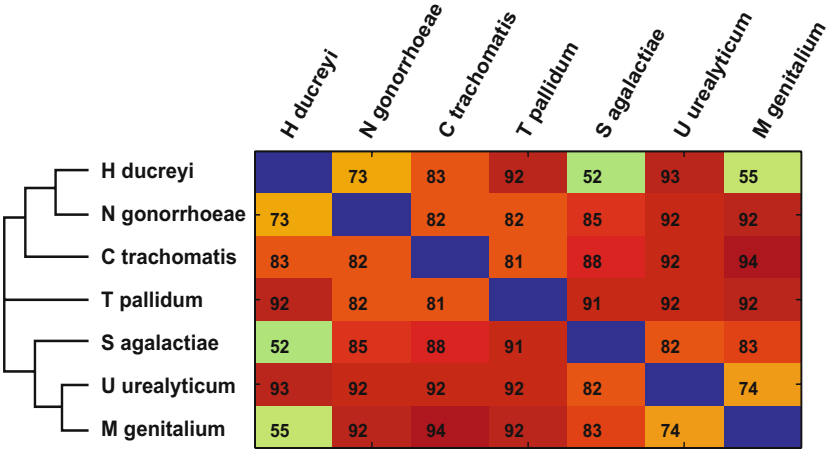


Fig. 1. Pairwise species discrimination performances. The AUC for classifiers trained to distinguish between proteins from each species pair (median of five replicates). The unrooted tree to the left shows the phylogenetic relationships of the seven bacterial species, based on 16S rRNA analysis.

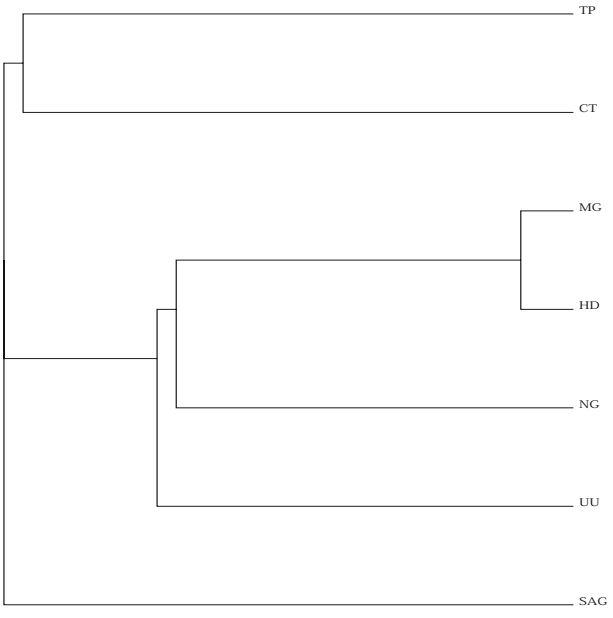


Fig. 2. Dendrogram resulting from pairwise species discrimination using SVMs. The median of the five AUCs of each pairwise species comparison was used as a distance measure between the species. See Table 1 for an explanation of the abbreviations.

are biologically very similar, they all occur in the same uniform environment, inside the urogenital tract of the human body. They naturally differ in their mechanisms of pathogenicity, but nonetheless their general biology should be the same and should make use of very similar molecular structures. The fact that we can identify general species-specific "sequence signatures" is therefore particularly striking.

In addition to the species–species discrimination, it was interesting to explore whether using sequence features to discriminate between bacterial species by machine learning will provide an accurate phylogenetic relationship between the species. The tree could then be compared to the 16S rRNA phylogenetic tree of the STD bacteria.

The median of the five AUCs of each pairwise species comparisons was used as a distance measure between the species in the phylogenetic tree. The OC [12] hierarchical cluster program was used to construct the dendrogram. For the 16S rRNA tree (Table 3), 16S rRNA sequences were obtained from GenBank for representative members of each bacterial genus in the dataset, as well as for three diverse Archaea for use as an outgroup. Sequences were aligned by ClustalW [16], positions containing gaps were removed, and the remaining alignment subjected to phylogenetic analysis using Maximum Likelihood (DNAml and Fast DNAml), Maximum Parsimony (DNAPars) and Neighbor Joining (DNADist+Neighbor),

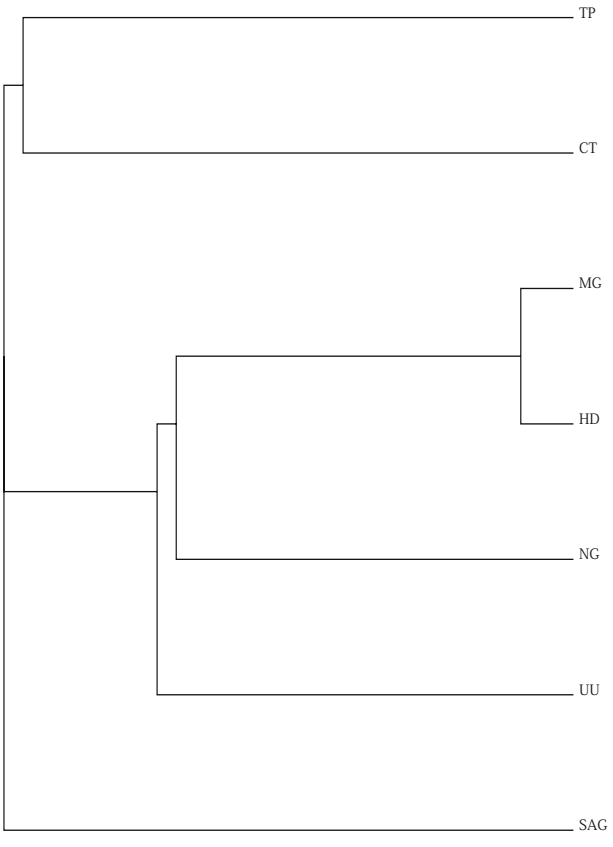


Fig. 3. 16S rRNA Dendrogram resulting from 16S rRNA sequences for the seven bacteria in our database as well as three Archaea outgroups. New species introduced as other members of the bacterial genus are NM = *Neisseria meningitidis* (as an alternative to NG) TD = *Treponema denticola* (as an alternative to TP) and US = *Ureaplasma parvum* serovar (as an alternative to UU). The Archaea include AF = *Archaeoglobus fulgidus*, HS =*Halobacterium salinarum* and PH = *Pyrococcus horikoshii*. see Table 1 for explanations of the other abbreviations.

using BioEdit. The resulting trees were identical except for a single change in the NJ tree. A majority-vote consensus tree was generated and rooted using the Archaea as outgroup.

From Figure 2 we can see that the SVM-based tree shows little similarity to the phylogenetic tree. For instance, the closely related species pairs *Haemophilus ducreyi*/*Neisseria gonorrhoeae* and *Mycoplasma genitalium*/*Ureaplasma urealyticum* are not identified correctly. Only *Chlamydia trachomatis* and *Treponema pallidum* are detected as outliers. However, we have generated random trees and found that the tree in Figure 2 is indeed closer to the 16S RNA tree than the majority of random trees, although this similarity is not obvious based on visual

inspection. This indicates that there is some useful phylogenetic signal contained in the SVM results. Perhaps more discriminatory sequence properties could be used for a more accurate construction of the dendogram.

4 Discussion

Comparing protein sequence properties of every species might outline the natural difference of the species and how evolution has played an essential role in their divergence. In this paper, we have used a wide range of sequence properties and have used Support Vector Machines in an attempt to discriminate between proteins of different species. Interestingly, the discrimination performances was as high as 85% AUC (median of all species–species discriminations performed). This is of course of great biological interest. By extracting useful information from the sequence, we hope to shed more light on this variation. At the DNA level, one of the discriminating species-specific features is the varying GC content. Guanine-cytosine (GC) content has been shown to be a biologically important attribute in prokaryotes [13]. It is known to be fairly balanced and tightly controlled across the genome, thus providing high specificity for genome identification. The Percentage GC content in bacteria can range from 25% to 75%. According to Bentley and Parkhill [13], the GC content of prokaryotes depends on the genome size. The correlation between genome size and GC content shows larger genomes tend to have higher GC content than smaller genomes which are AT-rich.

At the amino acid level, we expect preferred amino acids to be different as a result of varying GC contents at the DNA level. This is supported by earlier reports [17] which showed that the amino acid composition of 59 bacterial species was greatly influenced by varying genomic G+C content.

To further elaborate on this, we have recorded the highest 10 selected features when comparing the (GC-rich) β-proteobacterium *Neisseria gonorrhoeae* with the three (GC-poor) Firmicutes species (*Mycoplasma*, *Streptococcus*, and *Ureaplasma*). This is shown in Table 2.

The three amino acids alanine, arginine and proline have high GC content in most of their codons. Our features selected for classification of the proteins agree with the GC-based prediction, in that four features with enriched GC bases were selected amongst the top 10 discriminatory features. This is statistically highly significant and indicates the relevance of varying GC content for our successful species discrimination.

Further analysis of Table 2 shows the frequent occurrence of the amino acids lysine (Lys) and arginine (Arg) as relevant features. These amino acids have been reported earlier to be significantly overrepresented in proteins of particular functional categories (transcription, translation), indicating the importance of our selected features for protein function [18].

Finally, our method has demonstrated the discriminatory power of Support Vector Machine classification as it can use sequence features to discriminate proteins from different species with high reliability and accuracy.

Table 2. Top 10 selected features using the *Neisseria gonorrhoeae* genome. The top 10 selected features for comparing the β-proteobacteria *Neisseria gonorrhoeae* with the three Firmicutes species are shown. The amino acids that contain GC-rich codons are highlighted in bold. Abbreviations: AAs = amino acids, Qt = quarter, div = divided and no. = number. See text for discussion.

MG vs. NG	NG vs. SAG	NG vs. UU
The no. of amide AAs div by the length of the protein	The no. of mean blocks of charged AAs	The no. of tiny AAs div by the length of the protein
The no. of ile AAs div by the length of the protein	The no. of polar AAs div by the length of the protein	The no. of **pro** AAs from the 0% to 50% region of the protein div by the length of the protein
The no. of lys AAs	The no. of ile AAs div by the length of the protein	The no. of **arg** AAs in the protein
The no. of lys AAs div by the length of the protein	The no. of lys AAs in the 1st Qt of the protein div by the length of the protein	The no. of ile blocks in the 4th Qt of the protein div by the length of the protein
The no. of **arg** AAs in the 1st Qt div by the length of the protein	The no. of **ala** AAs div by the length of the protein	The no. of lys AAs in the longest lys block of the protein
The no. of mean blocks of **ala** AAs in the 3rd Qt of the protein	The no. of **pro** AAs div by the length of the protein	The no. of +ve charged AAs in the 1st Qt of the protein div by the length of the protein
The no. of **pro** blocks in the 25% to 75% region of the protein div by the length of the protein	The no. of gly AAs div by the length of the protein	The no. of **pro** AAs from the 50% to 100% region of the protein div by the length of the protein
The no. of mean blocks of **ala** AAs in the 2nd Qt of the protein	The no. of **arg** AAs in the 4th Qt of the protein div by the length of the protein	The no. of amide blocks in the 4th Qt of the protein div by the length of the protein
The no. of mean blocks of lys AAs in the 1st Qt of the protein	The no. of **arg** AAs in the protein	The no. of **pro** mean blocks in the protein
The no. of mean blocks of lys AAs in the 2nd Qt of the protein	The no. of cys AAs div by the length of the protein	The no. of ile mean blocks in the protein

We hope that this work can be extended by exploring further sequence properties as well as more diverse organisms, to elucidate the underlying biophysical and evolutionary mechanisms.

References

1. Zuckerkandl, E., Pauling, L.: Evolutionary divergence and convergence in proteins. In: Evolving Genes and Proteins, pp. 97–166. Academic Press, New York (1965)
2. Robichaux, R.H., Purugganan, M.D.: Accelerated regulatory gene evolution in an adaptive radiation. Proc. Natl. Acad. Sci. USA 98, 10208–10213 (2001)
3. Gollery, M., Harper, J., Cushman, J., Mittler, T., Girke, T., Zhu, J.-K., Bailey-Serres, J., Mittler, R.: What makes species unique? The contribution of proteins with obscure features. Genome Biology 7, R57 (2006)

4. STDGEN, Los Alamos National Laboratory Bioscience Division STD Sequence Databases, http://www.stdgen.lanl.gov
5. Ouali, M., King, R.D.: Cascaded multiple classifiers for secondary structure prediction. Prot. Sci. 9, 1162–1176 (2000)
6. Krogh, A., Larsson, B., von Heijne, G., Sonnhammer, E.L.L.: Predicting Transmembrane Protein Topology with a Hidden Markov Model: Application to Complete Genomes. J. Mol. Biol. 305, 567–580 (2001)
7. Linding, R., Jensen, L.J., Diella, F., Bork, P., Gibson, T.J., Russell, R.B.: Protein disorder prediction: implications for structural proteomics. Structure 11, 1453–1459 (2003)
8. Al-Shahib, A., Breitling, R., Gilbert, D.: Feature Selection and the Class Imbalance Problem in Predicting Protein Function from Sequence. Applied Bioinformatics 4, 195–203 (2005)
9. Al-Shahib, A., Breitling, R., Gilbert, D.: FrankSum: new feature selection method for protein function prediction. Int. J. Neural Syst. 15, 259–275 (2005)
10. Al-Shahib, A., Breitling, R., Gilbert, D.: Predicting protein function by machine learning on amino acid sequences – a critical evaluation. BMC Genomics 8, 78 (2007)
11. WEKA machine learning package, http://www.cs.waikato.ac.nz/ml/weka
12. Barton, G.: A cluster analysis program (1993), http://www.compbio.dundee.ac.uk/Software/OC/oc.html
13. Bentley, S.D., Parkhill, J.: Comparative Genomic Structure of Prokaryotes. Annual Review of Genetics 38, 771–791 (2004)
14. Shigenobu, S., Watanabe, H., Hattori, M., Sakaki, Y., Ishikawa, H.: Genome sequence of the endocellular bacterial symbiont of aphids Buchnera sp. APS. Nature 407, 81–86 (2000)
15. Bishop, M.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford (1993)
16. Chenna, R., Sugawara, H., Koike, T., Lopez, R., Gibson, T.J., Higgins, D.G., Thompson, J.D.: Multiple sequence alignment with the Clustal series of programs. Nucleic Acids Res. 31, 3497–3500 (2003)
17. Lobry, J.R.: Influence of genomic G+C content on average amino-acid composition of proteins from 59 bacterial species. Gene. 205, 309–316 (1997)
18. Bharanidharan, D., Gautham, N.: Amino acid variation in cellular processes in 108 bacterial proteomes. Arch. Microbiol. 184, 168–174 (2005)

Automatic Prognostic Determination and Evolution of Cognitive Decline Using Artificial Neural Networks

Patricio García Báez[1], Carmen Paz Suárez Araujo[2], Carlos Fernández Viadero[3], and José Regidor García[2]

[1] Departament of Statistics, Operating Reseach and Computation, University of La Laguna, 38071 La Laguna, Canary Islands, Spain
pgarcia@ull.es

[2] Institute of Cybernetic Sciences and Technology, University of Las Palmas de Gran Canaria, 35017 Las Palmas de Gran Canaria, Canary Islands, Spain
cpsuarez@dis.ulpgc.es

[3] Unidad de Atención a la Dependencia de Santander, Gobierno de Cantabria, 39012 Santander, Cantabria, Spain

Abstract. This work tries to go a step further in the development of methods based on automatic learning techniques to parse and interpret data relating to cognitive decline (CD). There have been studied the neuropsychological tests of 267 consultations made over 30 patients by the Alzheimer's Patient Association of Gran Canaria in 2005. The Sanger neural network adaptation for missing values treatment has allowed making a Principal Components Analysis (PCA) on the successfully obtained data. The results show that the first three obtained principal components are able to extract information relating to functional, cognitive and instrumental sintomatology, respectively, from the test. By means of these techniques, it is possible to develop tools that allow physicians to quantify, view and make a better pursuit of the sintomatology associated to the cognitive decline processes, contributing to a better knowledge of these ones.

1 Introduction

The progressive aging of the world population has supposed an increase in the incidence of diseases associated to the age. Among them, the neurodegenerative processes and dementia in particular have acquired a special relevance. Its high prevalence, 8-10% in older than 65 years and more than 25% in the very old ones [1], originates very important repercussions, not only in the scope of the patient, affecting to its autonomy and quality of life, but also in the familiar, social and, of course, sanitary spheres [2] [3].

These consequences urge us for the accomplishment of possible precocious diagnosis, and the most accurate as possible, and trying to resist to the maximum the devastating effects of this pathology by means of the accomplishment of a therapeutic plan and a suitable pursuit [4]. For this reason, the early detection

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 898–907, 2007.

of the cognitive decline is an advisable practice in all the scopes of attention. Nevertheless, the high uncertainty diagnosis [5], the degree of infradiagnosis, that can be so important that reaches 95% in some scopes, [6], advise to develop detection programs, new instruments of diagnose and guiding for the sanitary professionals. These actions have demonstrated their effectiveness as much in the attitude of the doctor on these pathologies like in evident improvements in the process of detection, diagnosis and treatment of the old ones affected by the dementia [7].

The evaluation of cognitive decline can be made by opened, semi structured form, or by means of the application of a series of cognitive evaluation scales. We must remember, nevertheless, that a test or a cognitive-based scale does not constitute a deterioration or normality diagnosis, but it only indicates the possible existence of a cognitive deficit. Throughout the years, there have been developed and have been used different neuropsychological tests [8] that evaluate different cognitive sections of a patient, but does not turn out simple to relate these tests to concrete sintomatologies or different levels of cognitive decline. It is necessary that they have been widely validated and contrasted and, of course, to know which is its level of sensitivity, specificity and predictive value. Other non-despicable problems in the use of these scales are the absence of universal cut points, the trans-cultural problematic and the precision that seems similar as much with the use of short and/or long scales [7] [9].

It is an usual practice to put patients under different tests. Their scores are strongly collinear, because the different tests are composed of common cognitive components. In order to approach this problem and to advance towards the diagnosis we present a proposal based on neural computation. Thus, using an artificial neural network to carry out a PCA, we can approach the inherent multicolineality between the different tests. This way, we obtain more accurate information about the patient state, which allows us to make prognoses and studies of the evolution of the cognitive decline.

2 Data Environment

The dataset is made of the results of the 267 clinical consultations made over 30 patients during 2005 at Alzheimer's Patient Association of Gran Canaria. Its structure includes a patient identifier, resulting of 5 neuropsychological tests and diagnose of cognitive decline (DCD) as well as diagnose of differential dementia. These data has as advantage that they are very homogeneous; each patient has scores of his/her monthly-made tests, excepting the Mini Mental test which is made twice a year. But, although the majority of the patients have been tested 12 times, there are some patients with fewer consultations made and some of data missing in the consultations. Thus, we work with a dataset where the missing data feature is present, neither in all the consultations nor all the patients are applied the complete test battery.

The 5 different used data tests are the following ones:

1. Mini Mental Status Examination (MMSE) [10] is the most spread, employed and quoted standardized instrument to value the cognoscitive function. It consists of a set of short and simple questions that allow a quick evaluation of several cognitive areas: orientation, fixation, calculation and attention, memory, language, reading, writing and visoconstructive abilities. Its score ranges from 0 to 30 points. It also constitutes the most used pruning test on the international epidemiological investigation, as well as in the clinical tests that require evaluating the patient's intellective functions.
2. FAST scale (Functional Assessment Staging) [11] is used to evaluate the possible relation between functional stage and survival. It consists of 7 stages (1 to 7) being the very last two ones, severe dementias and advanced ones, subdivided (6A to 6E and 7A to 7F). Derivated of our experience, it seems that the evaluation by means of a specific scale as FAST, in which the function is included, clearly orientates on the different stages towards a less accurate diagnose while the functional affectation rises [12]. A nondespicable disadvantage is derived from the generalization of the FAST scale, initially designed for patients of Alzheimer's, and therefore it does not have a sufficiently contrasted use in other dementias.
3. Katz's index [13] evaluates the pure function in the basic activities of the daily life: bath, dress, cleanliness, mobilization, continence and food. It is defined as an observation instrument and an objective guide of the course of the chronic disease, as an aid to study the process of aging and as an aid in the rehabilitation. Its score ranges from A to H.
4. Barthel's index [14] is similar to the one of Katz's with the difference of a numerical result, which is adapted for a continuous gradual evaluation, since the Katz's index is a ordinal scale with items of dichotomizing character. Like Katz's index, it evaluates the same functions this one does, although in a wider way. We use a summary score that consists of 5 stages: Independent, Slight, Moderate, Severe and Totally dependent.
5. Lawton-Brody's index [15] evaluates the behavior aspects of instrumental character, for which it is necessary to be able to make the basic functions of a suitable form. It implies therefore a much greater and more complex functional integrity. The use of the telephone, the capacity to make purchases, being able to cook, the house care, washing the clothes, the use of transport means, being able to handle the proper medication and the handling of the financial aspects are evaluated. Its score ranges from 0 to 8 points for women, and 0 through 5 for men.

The missing data corresponding to the MMSE test have been completed, in agreement with the clinical experts, by means of interpolation from the annual results of both annual tests that almost all the patients have been put under. Even so, other values have left empty, 71 of the total of 1335, whose distribution by test and number of patients is indicated in Table 1.

In order to facilitate the convergence, as previous step to their use, the different fields that constitute the successfully obtained information were

Table 1. Statistics of missing data in data set

$Test type$	$Number of missings data$	$Number of patients$
$MMSE$	36	4
$FAST$	0	-
$Katz$	33	12
$Barthel$	1	1
$Lawton - Brody$	1	1
$Total$	71	16

preprocessed. The neuropsycological tests were standardized between 0 and 10 from the minimum and maximum values that can be reached in these tests. Those fields not being filled up labeled as lost values or missing ones, had a later special treatment.

At a control stage, the values of diagnosis are used. The diagnosis of the CD can contain four different values: Without CD, Slight CD, Moderate Cd and Severe CD. From the entire set of consultations, 15% were diagnosed as Slight CD, 34% as Moderate CD and 51% as Severe CD, due to the data source does not include patients without cognitive decline. In the diagnosis of dementias, 73% of consultations correspond to Alzheimer-type dementias, being the other distributed 27% in other types of dementias.

3 Neural Approximation

The dimensionality and the data type to be treated make it difficult to observe the existing differences between different patients, or the differences that can be between diverse consultations of a patient. This is, in part, due to the co-linearity of the tests: some of the tests in fact measure diverse aspects or several tests measure similar components. In order to be able to approach this difficulty, we transformed the representative data of the different consultations into a more reduced dimensionality data space so that a minimum loss of information exist, and where each dimension represents features that have greater independence from the other ones and which can be assigned a concrete meaning to.

Projective methods for feature extraction generate a transformation of the input space into a new feature space with a lower dimensionality. Usually, this transformation is created on the basis of linear combinations that try to maximize some interesting measurement, like preserving the max possible amount of information, reducing remarkably the data dimensionality. Its utility is focused into the data compression or optimal codification, and the display of high dimensionality data. Some of the processes that make this reduction are the related to PCA. Among the neural approaches whose processes converge into PCA we have the Sanger network, also called Generalized Hebbian Algorithm (GHA) [16], the Adaptive Principal-component Extractor (APEX) [17], the Foldiak's model [18] and the Rubner's model [19]. A detailed comparative of PCA neural networks can be found in [20]. These processes allow generating a hierarchical structure

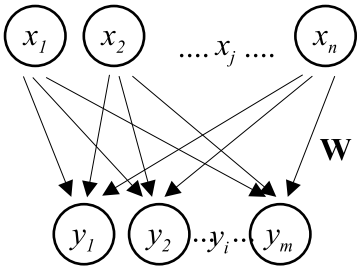


Fig. 1. Topology of Sanger neural network

of each of the new coordinates so that these are ordered based on the amount of information that represent on the total of data. Analyzing the existing relationship between the original coordinates space and the new coordinates space we can give meaning to each of the main components, this way is possible to represent each consultation by means of a reduced number of features that qualify the state of the patient at the moment of accomplishment of the consultation. The comparison between these features for different types of patients or consultations at different dates for a same patient can bring to light important information related to the prognosis and temporary evolution of the cognitive decline from the examined patients.

Using the Sanger network, there have been extracted and analyzed the first three main components of our data. As the shown in Figure 1, Sanger network is totally interconnected with the inputs; it is a linear processing network where the i-th unit has the y_i output that is given by the following expression:

$$y_i = \sum_{j=1}^{n} w_{ij} x_j \tag{1}$$

where n is the dimensionality of the input vector, x_j is the value of j-th input and w_{ij} is the weight between the j-th input and the i-th unit. The learning rule that progressively fits this matrix of weights is given by the following expression:

$$\Delta w_{ij} = \eta(t) y_i \left(x_j - \sum_{k=1}^{i} y_k w_{kj} \right). \tag{2}$$

In the expression given by Equation (2), the learning ratio $\eta(t)$ is a linearly decreasing function in time t. This formula causes the Sanger weights to converge to the main training data set components.

An important aspect for our developments is that the compression process carried out by this network is reversible, the original input vector ($\mathbf{x}$) can be reconstructed ($\mathbf{Rx}$) with a minimum loss of information from the m-outputs

vector ($\mathbf{y}$) and the Sanger weights ($\mathbf{W}$) (decompression procedure). The equation that carries out the decompression is the following one:

$$Rx_j = \sum_{i=1}^{m} w_{ij} y_i. \tag{3}$$

3.1 Sanger Network Extension for Missing Data Treatment

The previous expressions of the Sanger network must be adapted to be able to face the processing of missing data, to accomplish it we have followed a scheme similar to the one found in [21], causing these missing values not to contribute the output nor modify the weights of the network. This way, Equation (1) will be as it follows:

$$y_i = \sum_{j \in P_t}^{n} w_{ij} x_j \tag{4}$$

where P_t is the set of input units, j, whose values x_j are available at time t. Also, Equation (2) will be modified the following way:

$$\Delta w_{ij} = \begin{cases} \eta(t) y_i \left(x_j - \sum_{k=1}^{i} y_k w_{kj} \right) & \text{if } j \in P_t; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

This way, once the learning has ended, and having used the decompression procedure, it is possible to make an estimation of the value of the missing data.

4 Results and Discussion

4.1 Patients Prognostic

Once the Sanger network has been trained with our data set, we obtain the values for each one of the first three principal components in relation to the five tests, in Figure 2 are shown the obtained values for each one of the first three main components in relation to the five tests. We see that the first component depends on a great extent on the values of the Katz's and Barthel's tests, which takes us, according to the meaning of these tests, to the conclusion that this component represents the functional sintomatology. The second component is inversely proportional to the value of MMSE test, and directly proportional, but to a lesser extent, with the FAST test. It indicates to us that this second component is strongly related to the cognitive sintomatology. Finally, the third component depends mainly on the value of the Lawton-Brody test, and to a lesser extent of the value of Barthel test and inversely to the value of the MMSE. The representation of this component is then bound to the instrumental sintomatology that features the Lawton-Brody test. It is possible to observe therefore that the extraction of main components can isolate the main characteristics that are mixed in the different tests.

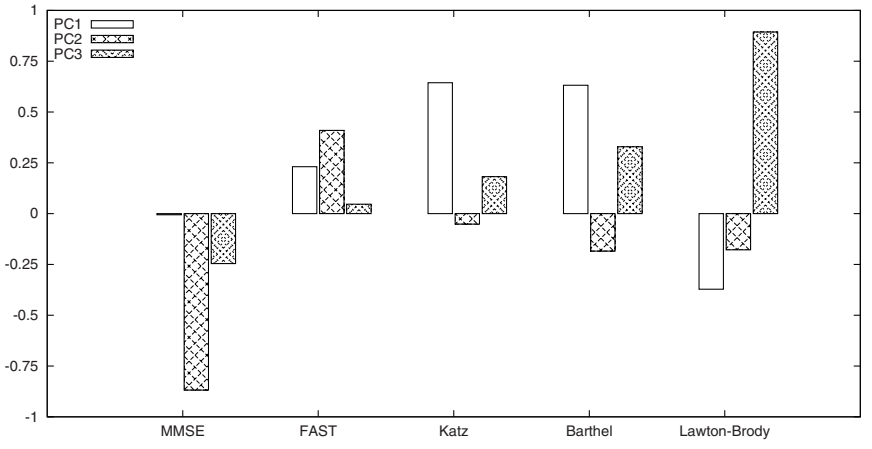


Fig. 2. Values of first three principal components

In Figure 3a, it can be seen data of all the consultations facing the first against the second main components. In addition, each consultation has been labeled with the patient's DCD at that moment. The CD is clearly represented within the second component, grouping the most of the slight CD below value -3, moderate CD between -3 and 1 and severe CD over 0. On the other hand, the functional skills of the patients are represented by the first component, having those patients with greater functional problems higher values. There exists a relationship between this component and DCD, since habitually patients with greater deterioration have more diminished their functional skills. Thus, we see in the graph that most of patients with slight CD or moderate CD have negative values of this component, having positive values of the most of patients with severe CD.

Figure 3b represents the second against the third components. The analysis carried out by the clinical experts on these data verifies to us the relationship of the third main component with the instrumental skills of patients. In the case of this component, a direct relationship with DCD is not appraised, since the different groups distribute themselves the same in their rank.

4.2 Temporary Evolution of Patients

Once our data has been processed by means of the Sanger network, we obtain for each of the consultations a point of the new coordinates space given by the values of each of the principal components that conform it. If we unit the points of a same patient from the first to the last consultation we will have a path in this space, this path will indicate us the temporary evolution that the patient has had, that is to say, the trajectory that follows between each point indicates for each coordinate if there have been improvements or worsenings in the sintomatology associated to this coordinate.

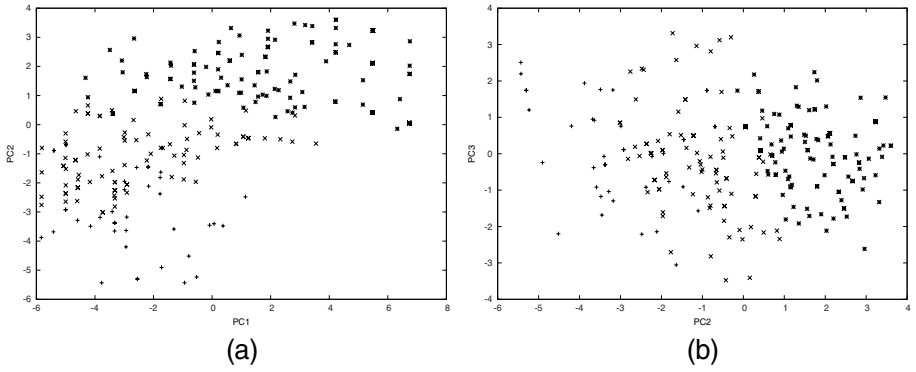


Fig. 3. Graphical projection of (a) scores of first and second factors from the PCA and (b) scores of second and third factors from the PCA. + show patients with slight CD, x with moderate CD and $*$ with servere CD.

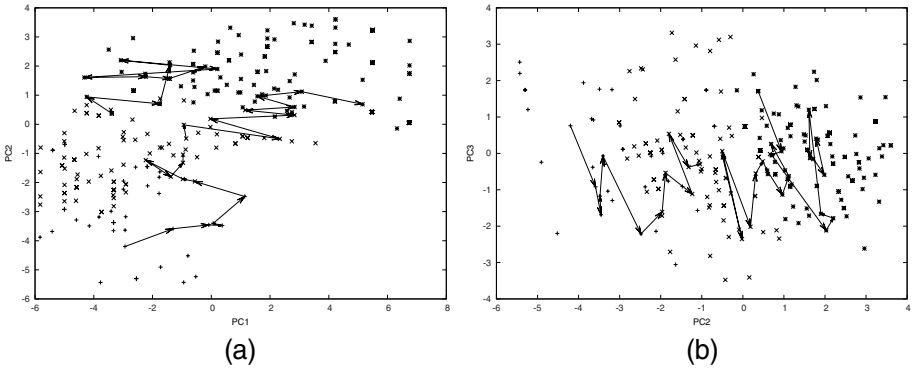


Fig. 4. The arrows show the temporal evolution of (a) scores of first and second factors from the PCA and (b) scores of second and third factors from the PCA of three different patients

In Figure 4, we can observe the traces corresponding to the consultations associated to three different patients. Two of them change of state of moderate CD to severe CD, and the third one starts from slight CD and reaches the moderate CD. It is possible to observe that throughout the trajectories the appearance of oscillations is frequent, according to our clinical team these changes are in agreement with the real evolutionary situation of the patients. The decline processes are usually not stable, but patient can undergo small improvements or worsenings throughout which can be caused by diverse external factors.

It is possible that the maximum passed time between the first and last consultation, a year, turns out short to be able to see the global evolution of the patient with clarity over the commented oscillations in the data we have. Despite, indeed the utilities of this analysis relative to the pursuit of the evolution

of the disease seem clear, comparison between evolutions of different patients and even prognosis from paths that could follow in the future.

5 Conclusions

The present work supposes an important advance in the area of the neuroinformatics and medical computer science, since it proposes new techniques of automatic learning to analyze and process data relative to neurodegenerative processes habitually related to the aging. On the basis of its results the creation of tools that allow the doctors to quantify, visualize and make a better pursuit of the associated sintomatology to these processes will be possible, contributing to a better knowledge of these ones.

In clinical practice, it is usual not to have the required clinical tests to evaluate a patient. In order to be able to face this difficulty in the PCA accomplishment we have extended the Sanger network equipping it with capacities for processing missing data.

PCA on subjects that present CD has allowed us to dissect the inherent multicolinearity from the different results of neuropsychological tests. We have been able to quantify in an independent way for each subject the functional, cognitive and instrumental sintomatologies that this one presents, and that is expressed by the different associated values from the first three principal components.

The analysis of the temporary evolution of the main components values has been like a powerful tool for the pursuit of patients. Allowing to detect the different sintomatologic changes and to compare these changes with the produced ones in the rest of patients. this analysis could allow us in a future to categorize the different evolutionary pictures, as well as to orient the doctor onto the possible prognosis that can experience his patients sintomatology. Our proposal will also make it easier to prognose the degeneration in a dementia as well as to make studies on pharmacological treatments, all which will result in the possibility of reaching one better quality of life of the patient.

Acknowledgement

We wold like to thank to Gobierno de Canarias for their support in de undertaking of this research under Research Project PI2003/178A and to Alzheimer's Patient Association of Gran Canaria for their generous input as data provider.

References

1. López-Pousa, S.: La demencia: concepto y epidemiología. Enfermedad de Alzheimer y otras demencias. Médica Panamericana, Madrid , 33–42 (2006)
2. Andersen, C.K., Wittrup-Jensen, K.U., Lolk, A., Andersen, K., Kragh-Sorensen, P.: Ability to perform activities of daily living is the main factor affecting quality of life in patients with dementia. Health Qual Life Outcomes 2 (2004)

3. DeKosky, S.: Early intervention is key to successful management of Alzheimer disease. Alzheimer Disease and Associated Disorders 17, S99–S104 (2003)
4. Doody, R.S.: Refining treatment guidelines in Alzheimer's disease. Geriatrics Suppl. 14–20 (2005)
5. Neuropathology Group of the Medical Research Council Cognitive Function and Ageing Study: Pathological correlates of late-onset dementia in a multicentre, community-based population in England and Wales. The Lancet 357, 169–175 (2001)
6. Solomon, P., Murphy, C.: Should we screen for Alzheimer's disease? Geriatrics 60, 26–31 (2005)
7. Borson, S., Scanlan, J.M., Watanabe, J., Tu, S.P., Lessing, M.: Simplifying detection of cognitive impairment: comparison of the Mini-Cog and Mini-Mental State Examination in a multiethnic sample. Journal of the American Geriatrics Society 53, 871–874 (2005)
8. Nestor, P.J., Scheltens, P., Hodges, J.R.: Advances in the early detection of Alzheimer's desease. Nature Reviews Neuroscience 5, S31–S34 (2004)
9. Damian, M., Kreis, M., Krumm, B., Hentschel, F.: Optimized neuropsychological procedures at different stages of dementia diagnostics. Journal of the Neurological Sciences 229, 95–101 (2005)
10. Folstein, M.F., Folstein, S.E., McHugh, P.R.: Mini-mental state. A practical method for grading the cognitive state of patients for the clinician. Journal of psychiatric research 12, 189–198 (1975)
11. Reisberg, B.: Functional Assessment Staging (FAST). Psychopharmacology Bulletin 24, 653–659 (1988)
12. Fernandez-Viadero, C., Verduga, R., Crespo, D.: Biomarcadores biológicos del envejecimiento. Biogerontología, Universidad de Cantabria, Cantabira (2006)
13. Katz, S.C., Ford, A.B., Moskowitz, R.W.: Studies of illness in the aged. The index of ADL: a standardized measure of biological and psychosocial function. JAMA 185, 914–919 (1963)
14. Mahoney, F.I., Barthel, D.: Functional evaluation: The Barthel Index. Maryland State Medical Journal 14, 61–65 (1965)
15. Lawton, M.P., Brody, E.M.: Assessment of older people: self-mantaining and instrumental activities of daily living. Gerontologist 9, 179–186 (1969)
16. Sanger, T.D.: Optimal Unsupervised Learning in a Single-Layer Linear Feedforward Neural N. Neural Networks 2, 459–473 (1989)
17. Dimantaras, K., Kung, S.Y.: Principal Component Neural Networks: Theory and Applications. John Wiley and Sons, New York (1996)
18. Foldiak, P.: Adaptive network for optimal linear feature extraction. In: Proceedings of the International Joint Conference on Neural Networks I, pp. 401–405 (1989)
19. Rubner, J., Tavan, P.: A self-organizing network for principal component analysis. Europhysics Letters 10, 693–698 (1989)
20. Fiori, S.: An Experimental Comparison of Three PCA Neural Networks. Neural Processing Letters 11, 209–218 (2000)
21. Samad, T., Harp, S.A.: Self-organization with partial data. Network 3, 205–212 (1992)

SCSTallocator: Sized and Call-Site Tracing-Based Shared Memory Allocator for False Sharing Reduction in Page-Based DSM Systems*

Jongwoo Lee, Youngho Park, and Yongik Yoon

Dept. of Multimedia Science, Sookmyung Women's University, Seoul 140-742, Korea
{bigrain, yhpark, yiyoon}@sookmyung.ac.kr

Abstract. False sharing is a result of co-location of unrelated data in the same unit of memory coherency, and is one source of unnecessary overhead being of no help to keep the memory coherency in multiprocessor systems. Moreover, the damage caused by false sharing becomes large in proportion to the granularity of memory coherency. To reduce false sharing in page-based DSM systems, it is necessary to allocate unrelated data objects that have different access patterns into the separate shared pages. In this paper we propose *sized and call-site tracing-based shared memory allocator*, shortly *SCSTallocator*. SCSTallocator expects that the data objects requested from the different call-sites may have different access patterns in the future. So SCSTallocator places each data object requested from the different call-sites into the separate shared pages, and consequently data objects that have the same call-site are likely to get together into the same shared pages. At the same time SCSTallocator places each data object that has different size into different shared pages to prohibit the different-sized objects from being allocated to the same shared page. We use execution-driven simulation of real parallel applications to evaluate the effectiveness of our SCSTallocator. Our observations show that our SCSTallocator outperforms the existing dynamic shared memory allocators. By combining the two existing allocation technique, we can reduce a considerable amount of false sharing misses.

Keywords: False Sharing, Distributed Shared Memory, Dynamic Memory Allocation, Sized Allocation, Call Site Tracing.

1 Introduction

In distributed shared memory (DSM) systems, efficient data caching is crucial to the entire system performance due to the non-uniform memory access time characteristics. Because the access to a remote memory is much slower than the access to a local memory, reducing the frequencies of the remote memory accesses with efficient caching can lead to decrease of the average cost of memory accesses, and subsequently improve the entire system performance [1]. A simple and widely used

* This Research was supported by the Sookmyung Women's University Research Grants 2006.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 908–918, 2007.

mechanism for exploiting locality of reference is to replicate or migrate frequently used pages from remote to local memory [2]. But in case of page replication, the existence of multiple copies of the same page causes memory coherence problem (Fig. 1).

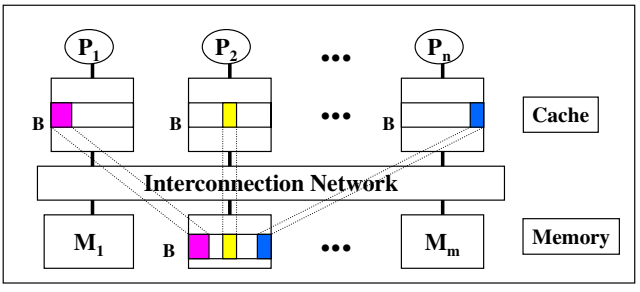


Fig. 1. Example of page replication in DSM systems

In DSM systems, ***false sharing*** happens when several independent objects, which may have different access patterns, share the memory coherency unit. Memory faults or misses caused by false sharing do not affect the correct executions of the parallel applications. As a result, we can say that false sharing is one major source of unnecessary overhead to keep the memory consistent [3, 4]. Especially, the problem becomes severe in PC-NOW DSM systems where the memory coherency unit is very large (generally, one virtual page). They find out that the false sharing misses occupy 80% or so of the shared memory faults in page-based DSM systems [5, 6]. It means that the false sharing is the major obstacle for improving the memory performance in page-based DSM systems. In this paper, we present an efficient dynamic shared memory allocator for false sharing reduction. The reasons why we chose to optimize dynamic shared memory allocator for reducing false sharing are that this approach is transparent to the application programmers, and almost all the false sharing misses happen in shared heap when multiple processes in a parallel application communicate with each other using shared memory allocated by dynamic shared memory allocator.

To reduce the false sharing misses in a shared memory allocator, a prediction of the future access patterns of each data object is necessary. To predict the future access pattern of each data object at the shared memory allocators run-time, several techniques are suggested [5, 6, 8, 9]. Among them, [5, 6, 8] and [9] motivated us to expect that, by combining the two techniques, more performance improvements can be possible. In [5, 6, 8] and [9], the sized allocation technique and call-site tracing based allocator(shortly CSTallocator) are proposed respectively. As we can see from the name of the techniques, sized allocator uses the request size as a clue to predict the future access pattern of each data object. The sized allocator expects that data objects of different sizes are likely to show different access patterns in the future. In CSTallocator, on the other hand, by tracing the call-site(object request location in parallel program codes), data objects requested at different locations should not be allocated in the same shared page. This is based on the idea that data objects requested at the different call-sites will show different access patterns in the future.

In section 2, we review the related works. Section 3 explains the design and implementation of the combined version (SCSTallocator). We present the results of performance evaluation in section 4, and section 5 draws the conclusions.

2 Related Works

In this paper, we focus on the page-based DSM systems that keep the memory coherency in unit of a virtual memory page. The dynamic shared memory allocator for the page-based DSM systems has to decide where the requested data objects are placed. If the dynamic shared memory allocator knows the characteristics and access patterns of the requested data objects in advance, the allocator can easily place the data objects into the appropriate shared page to blockade the false sharing misses. For example, the allocator can reduce the false sharing misses by placing the objects with much different access patterns to the different shared pages, or not placing non-related data objects into the same shared page. But, the dynamic shared memory allocator cannot know the characteristics and access patterns of the requested objects in advance. Therefore, the *typed allocation* is proposed in [7] where the clues provided by the programmers are used. In this typed allocation, the programmer must specify the memory access type through the allocation function arguments, such as Read-Only, Write-Mostly, and Lock types. Thus, the data objects with different types could be placed in the different shared pages. But, this scheme needs to additional overheads that user interfaces of the dynamic shared memory allocator have to be changed, and in turn the modification of the application source code is unavoidable. Moreover, it is not an easy job for the programmers to know in advance the access types of each shared data object. Our work assumes that there should be no changes in the API of the dynamic shared memory allocator.

Per-process allocation scheme assigns the different cache lines to the data objects requested by the different processes [3, 14, 15, 16]. In this scheme, the data objects requested by the different processes are placed in the separate cache lines, so that it could reduce the possibility that data objects without relationships or with different access patterns are placed in the same cache line. This technique is effective where multiple processes request shared memory allocation evenly, but is likely to be ineffective where a dedicated process has the full responsibility of shared memory allocation [8]. In all the parallel applications used in our experiments, a dedicated process is also used for shared memory allocation, so it is inappropriate to compare this scheme with our approach.

Sized allocator (Fig. 2) is proposed in [5, 6, 8], where the different-sized objects are prohibited from being placed in the same shared page. That is, by placing only the same-sized objects in the same shared page, this method tries to minimize the co-location of heterogeneous data objects. They say that, by using the object-size information for the prediction of the future access patterns, the transparency of the allocator API could be kept and the false sharing misses could be reduced simultaneously. But this sized allocation is not enough to exactly predict the future access patterns of the shared data objects because the object size may not sufficiently represent the future access patterns.

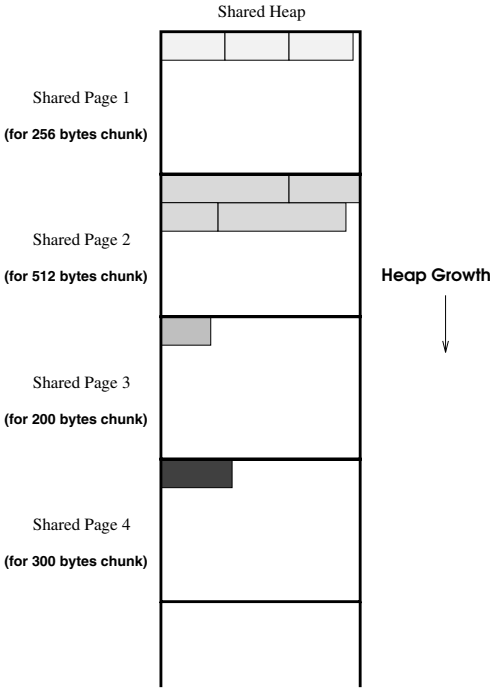


Fig. 2. Shared objects allocation example according to the object size in sized-allocation scheme

In [9], call-site tracing based shared memory allocator, shortly called *CSTallocator*, is presented(Fig. 3). In this technique, the future access patterns are predicted by the shared objects' request location(call-site) in the program codes. That is, the prediction is performed based on the instruction pointer from where the shared object allocation is requested. They hope that the objects with different call-sites may have the different access patterns in the future. By using the implicit information inherent in the program codes, CSTallocator tries not only to keep the API transparency, but also to unburden the programmers' efforts. There, authors claim that the call-site information of a shared object could be a useful clue for predicting the future access patterns because most parallel application programs call the allocation functions at different locations according to the object usage plans. Of course, they admit that the call-site tracing cost is more expensive than the cost of getting static information such as the request size passed via parameters or processor/process ID calling the function.

We find out by reviewing the previous works that the effective prediction of the future access patterns is a very important factor to reduction of the false sharing misses. And we can also find out that the sufficient false sharing reduction can't be obtained using just one prediction technique. These observations motivated our work to combine the existing prediction techniques(sized allocation scheme and CSTallocator) to simultaneously use them at the dynamic shared memory allocator run-time.

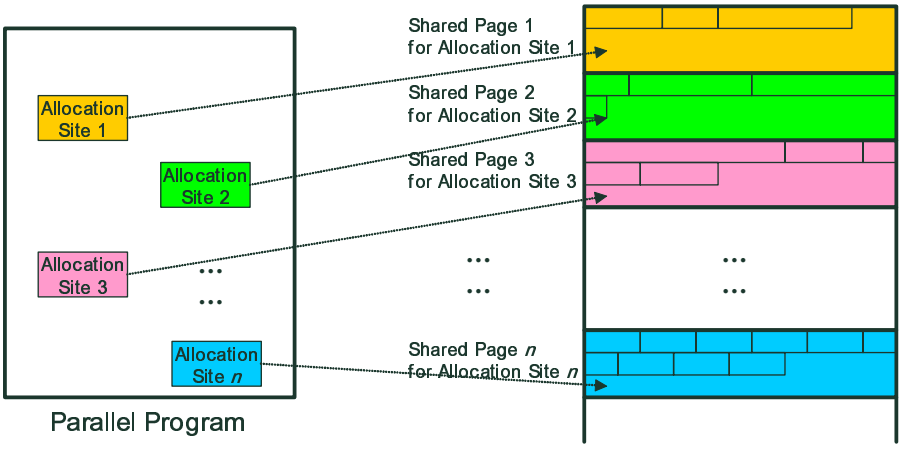


Fig. 3. Shared objects allocation example according to the call-sites in CSTallocator

3 *SCSTallocator:* Combining the *Sized* and *CSTallocator*

The object size and the call-site have a different point of view. Object sizes are syntactic information describing the characteristics of each object. We can say, on the other hand, call-sites are semantic information showing the future usage pattern of each object. We expect that, by combining the merits of the two heterogeneous prediction clues, larger amount of false sharing misses can be reduced. To evaluate the effectiveness of the combined version, we implement an allocator using the two prediction policies, sized allocation and CSTallocator, simultaneously. Fig. 4 shows an example of allocations when using this combined version. We are sure that the two prediction clues are the only information we can obtain from the allocators without modifying their API.

As we can see in this example, SCSTallocator allocates each object into the different pages if even one of the object size and call-site is different from each other. After all, a shared page contains only the same-sized objects requested at the same location in program codes hoping for that co-location of the different objects that may have different access patterns can be minimized. Because both the object size and the call-site are used to classify each object, the same-sized objects requested at the different call-sites are allocated into the different shared pages, and similarly the same call-site's objects of different size are also allocated into the different shared pages. The case SCSTalocator cannot cover is the following situation: future access patterns of the objects allocated into the same shared pages are unexpectedly different from each other. However, any one who has some experiences of parallel programming can easily accept that the above worst case situation may rarely happen because it is impossible to request several objects of different usage plans at the same call-site even though they have the same size.

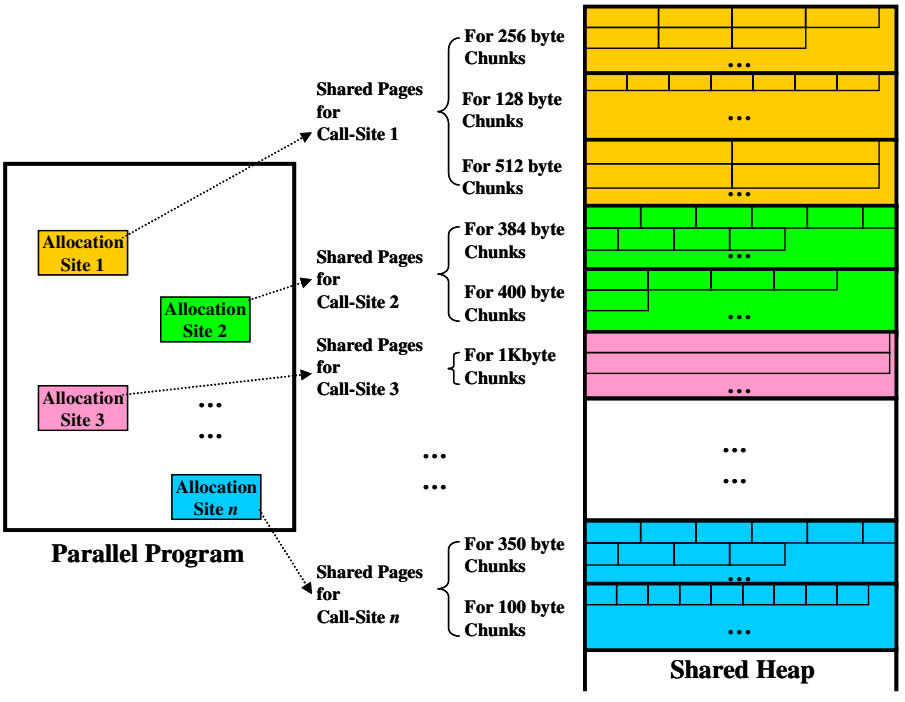


Fig. 4. An example of allocations when using the *SCSTallocator* (Sized allocation + CSTallocator)

Fig. 5 shows the implementation snapshot of bucket management in the SCSTallocator. Here, a bucket means a unique allocation control slot to gather only the homogeneous pages.

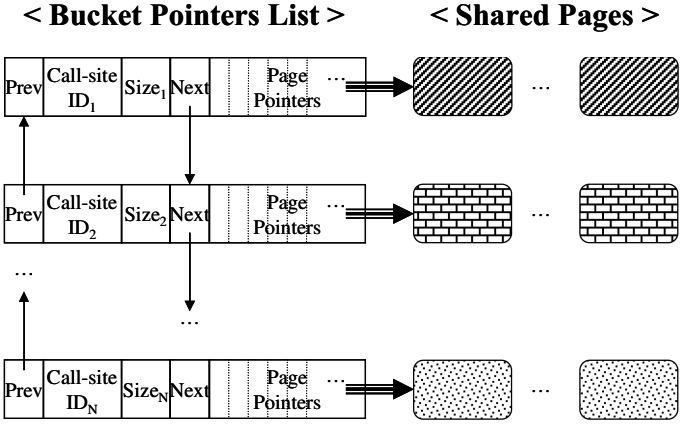


Fig. 5. Bucket management snapshop in SCSTallocator

4 Performance Evaluation

This section explains the experimental environments and shows the results of the false sharing misses measurement, comparing with the performance of the three allocators: CSTallocator, the sized allocator, and the SCSTallocator.

4.1 Experimental Environments

We use the execution-driven technique to simulate a DSM system consisting of 16 nodes. The simulator consists of the front-end and the back-end simulators. The front-end simulator interprets the execution codes of the parallel application program binaries and simulates the executions of the processors. We use MINT(Mips INTerpreter) [10, 11] as a front-end simulator. The back-end simulator simulates the policies of the memory management system using MINT's outputs. MINT interprets the execution codes and calls functions provided by the back-end simulator in every memory reference. The back-end simulator implements the memory management policies and the memory coherence protocols to be simulated.

We use cholesky, mp3d, barnes, and volrend as parallel application program suites. These parallel applications are randomly selected from the Stanford's SPLASH [12] and SPLASHII [13]. We compare the number of false sharing misses when using the three allocation schemes, CSTallocator, sized allocator, and SCSTallocator. Table 1 shows the parallel applications used in our simulation and their characteristics.

Table 1. The parallel applications used in simulation and their characteristics

Application's Name	Memory Reference Length (10^6)	Working Set Size (MB)	Application's Function
Cholesky	39.341	2.88	Cholesky factorize a sparse matrix
Barnes	58.716	1.20	Simulate evolution of galaxies
Mp3d	8.935	2.02	Simulate rarefied hypersonic flow
Volrend	31.186	3.91	3D rendering using ray tracing technique

4.2 Experimental Results

Table 2 shows how many false sharing misses are reduced in each parallel application when using each allocator. The number of buckets in the second column is the number of the unique allocation slots found during the repeated shared memory allocation function calls. It represents the number of unique object sizes when using the sized allocator, and the number of unique call-site IDs when using CSTallocator respectively. In case of the SCSTallocator, the number of buckets is equal to the number of unique (Call-Site ID, object size) pairs. These three allocators manage each object as a linked list using the separate pointers for its bucket. The shared pages with the same bucket pointers are assigned to the data objects with the same call-site ID or object size or (Call-Site ID, object size). Thus, the more buckets are found, the more sophisticated classification has been done.

According to our expectation, we can see from the Table 2 that the SCSTallocator outperforms the other two allocators. This means that the combined clue (object size + Call-Site ID) is much more effective in predicting the future memory access

patterns of data objects than only the single clue. And we can also find out that the more classification buckets we use the more accurate prediction result we can obtain.

Table 2. Results of performance comparison of sized allocation, CSTallocator, and the SCSTallocator (page size = 4KB)

(a) Cholesky

Measure / Allocator	# of buckets	# of false sharing misses	Reduction rate(%)
Sized	10	44,717	
CSTallocator	17	36,599	18.2
SCSTallocator	21	34,687	22.4

(b) Mp3d

Measure / Allocator	# of buckets	# of false sharing misses	Reduction rate(%)
Sized	8	6,147,589	
CSTallocator	5	5,754,143	6.4
SCSTallocator	11	5,312,768	13.6

(c) Barnes

Measure / Allocator	# of buckets	# of false sharing misses	Reduction rate(%)
Sized	27	5,805,705	
CSTallocator	7	5,104,413	12.1
SCSTallocator	29	4,814,970	17.1

(d) Volrend

Measure / Allocator	# of buckets	# of false sharing misses	Reduction rate(%)
Sized	11	953	
CSTallocator	12	883	7.3
SCSTallocator	17	798	16.3

A disadvantage when using our SCSTallocator is the additional space overhead caused by using the more buckets. The next section shows the analysis results of each allocator's space overheads.

4.3 Analysis of Space Efficiency

For the strict performance evaluation, we need to analyze the space overheads caused by CSTallocator, the sized allocator, and the SCSTallocator respectively. The space overhead is the amount of additionally used pages by each method.

At first, we analyze the general shared memory allocator, which does not use the buckets such as object size or call-site ID. In the general shared memory allocator, the objects are mixed up into the same shared page regardless of their sizes or call-sites. So the allocation requests stream, S, of a general allocator is represented as:

$$S = \{s_1, s_2, ..., s_n\} \tag{1}$$

where s_i = requested size of i-th allocation ($1 \le i \le n$), n = total # of requests.

The number of pages needed to accept the above allocation requests stream is as follows:

$$\#\text{ of pages required} = \left\lceil \frac{\sum_{i=1}^{n} s_i}{\text{page size}} \right\rceil \tag{2}$$

On the other hand, when using CSTallocator or sized allocator, or SCSTallocator, the allocation request stream can be represented as follows if the order of requests is ignored:

$$\begin{aligned} &S = \{S_{bucket_1}, S_{bucket_2}, ..., S_{bucket_k}\}, \\ &S_{bucket_k} = \text{set of allocations with bucket ID } bucket_k, \\ &S_{bucket_1} \cap S_{bucket_2} \cap ... \cap S_{bucket_k} = \varnothing, \\ &BS = \{bucket_1, bucket_2, ..., bucket_k\} : \text{set of unique bucket IDs.} \end{aligned} \tag{3}$$

And the number of pages needed to accept the above stream is as follows:

$$\#\text{ of pages required} = \sum_{bucket_k \in BS} \left\lceil \frac{|S_{bucket_k}| \times AvgSize_{bucket_k}}{\text{page size}} \right\rceil, \tag{4}$$

where $AvgSize_{bucket_k}$ is average size of each allocation request heading for $bucket_k$.

In comparison of the equation (2) with (4), the difference lies in the number of ceiling function. In equation (2), the ceiling function is applied at once, while it is applied as many as the size of the set BS (|BS|) in equation (4). This means that the maximum additional number of pages is limited to the number of the unique allocation sizes in sized allocation scheme, the number of call-site IDs in CSTallocator, and the number of unique (allocation size, call-site ID) pairs in the SCSTallocator respectively. Thus, the following is valid:

$$\text{Space Overhead} = \left(\sum_{bucket_k \in BS} \left\lceil \frac{|S_{bucket_k}| \times AvgSize_{bucket_k}}{\text{page size}} \right\rceil \right) - \left\lceil \frac{\sum_{i=1}^{n} s_i}{\text{page size}} \right\rceil \le |BS| \tag{5}$$

The obvious fact we can obtain from the equations (5) is that the shared page overhead is no more than the number of buckets regardless of which bucket classification methods are used. Table 3 shows the comparison results about the space efficiency measured by equation (5). As we can see in this table, the space efficiency

of CSTallocator is better than that of the sized allocator for mp3d and barnes, but is a little worse for cholesky and volrend. It means that the space efficiency gap between the two schemes is not quite large. Another thing we can see in Table 3 is that the space overhead of our SCSTalocator is not quite large when taking into account its performance improvements.

Table 3. Space efficiencies of the three allocators (Page size = 4KB)

Parallel application programs (total # of pages needed in general allocation method)	# of additional pages (space overhead (%))		
	Sized allocation	CSTallocator	**SCSTallocator**
Cholesky (738)	10 (1.36)	17 (2.30)	21 (2.85)
Mp3d (553)	8 (1.45)	5 (0.90)	11 (1.99)
Barnes (308)	27(8.77)	7 (2.27)	29 (9.42)
Volrend (441)	11 (2.49)	12 (2.72)	17 (3.85)

5 Conclusions and Future Works

This paper presents an efficient shared memory allocation method for parallel applications which communicate via dynamically allocated shared memory in DSM systems. Our allocator, called SCSTallocator, can reduce the false sharing misses more effectively than the two existing allocation schemes, sized allocator and CSTallocator. In SCSTallocator, both the object size and the call-site ID are used for predicting the future access patterns of each object. So only the same sized objects requested from the same location in program codes are allocated into the same shared pages. We are sure that the combined clue predicts the programmer's intention more accurately than the single clue, and find out by execution-driven simulation that the SCSTallocator outperforms the existing allocators. The SCSTallocator additionally spends pages only as many as the number of unique (object size, call-site ID) pairs. That is, our method can reduce more false sharing misses at the small sacrifice of space overhead. We expect that our SCSTallocator can contribute to both the reduction of false sharing misses and the reduction of the cost on keeping the memory coherency in DSM systems.

In the future, to measure the time efficiency as well as space efficiency, we will try to use the real DSM systems as a test-bed instead of simulation environments.

References

[1] Tanenbaum, A.S.: Distributed Operating Systems. ch.6, pp. 333–345. Prentice Hall, Englewood Cliffs (1995)

[2] Lee, J., Cho, Y.: Page Replication Mechanism using Adjustable DELAY Counter in NUMA Multiprocessors. J. Korean Institute of Telematics and Electronics B 33B(6), 23–33 (1996)

[3] Jeremiassen, T.E., Lam, M.S., Hennessy, J.L.: Shared Data Placement Optimizations to Reduce Multiprocessor Cache Miss Rates. In: ICPP 1990, vol. II(Software), pp. 266–270 (1990)
[4] Eggers, S.J., Jeremiassen, T.E.: Eliminating False Sharing. In: ICPP 1991, vol. I(Architecture), pp. 377–381 (1991)
[5] Lee, J., Cho, Y.: Shared Memory Allocation Mechanism for Reducing False Sharing in Non-Uniform Memory Access Multiprocessors. J. Korean Information Science Society(A): Computer Systems and Theory 23(5), 487–497 (1996)
[6] Lee, J., Cho, Y.: An Effective Shared Memory Allocator for Reducing False Sharing in NUMA Multiprocessors. In: IEEE 2nd ICA3PP 1996, pp. 373–382 (1996)
[7] Adema, R.L., Ellis, C.S.: Memory Allocation Constructs to Complement NUMA Memory Management. In: IEEE 3rd Symposium on Parallel and Distributed Processing (1991)
[8] Lee, J., Kim, M., Han, J., Ji, D., Yoon, J., Kim, J.: Effects of Dynamic Shared Memory Allocation Techniques on False Sharing in DSM Systems. J. Korean Information Science Society(A): Computer Systems and Theory 24(12), 1257–1269 (1997)
[9] Lee, J., Kim, S.D., Lee, J.W.: CSTallocator: Call-Site Tracing based Shared Memory Allocator for False Sharing Reduction in Page-based DSM Systems. In: 2nd Int. Conf. on High Performance Computing and Communications, pp. 148–159 (2006)
[10] Veenstra, J.E.: MINT: Tutorial and User Manual. Technical Report TR452, Computer Science Department, University of Rochester (1993)
[11] Veenstra, J.E., Fowler, R.J.: MINT: A Front End for Efficient Simulation of Shared-Memory Multiprocessors. In: 2nd Int. Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 201–207 (1994)
[12] Singh, J.P., Weber, W., Gupta, A.: SPLASH: Stanford Parallel Applications for Shared-Memory. ACM SIGARCH Computer Architecture News 20(1), 5–44 (1992)
[13] Woo, S.C., Ohara, M., Torrie, E., Singh, J.P., Gupta, A.: The SPLASH2 Programs: Characterization and Methodological Considerations. In: 22nd Annual Int. Symposium on Computer Architecture, pp. 24–36 (1995)
[14] Berger, E., McKinley, D., Blumofe, K.S., Wilson, R.D.: Hoard: A scalable memory allocator for multithreaded applications. In: 9th Int. Conf. on Architectural Support for Programming Languages and Operating Systems, pp. 117–128 (2000)
[15] Berger, E.D.: Memory Management for High-Performance Applications. PhD thesis, University of Texas at Austin (2002)
[16] Michael, M.M.: Scalable Lock-Free Dynamic Memory Allocation. In: ACM SIGPLAN 2004 Conf. on Programming Language Design and Implementation (2004)

A Hybrid Social Model for Simulating the Effects of Policies on Residential Power Consumption

Minjie Xu[1], Zhaoguang Hu[2], Xiaoyou Jiao[1], and Junyong Wu[1]

[1] School of Electrical Engineering, Beijing Jiaotong University, Beijing 100044, China
[2] State Power Economic Research Institute of SG, Beijing 100761, China
{xuminjie,huzhaoguang,jiaoxiaoyou}@chinasperi.com.cn

Abstract. In this paper, a hybrid social model of econometric model and social influence model is proposed to settle the problem in power resources management. And, a hybrid society simulation platform based on the proposed model, termed Residential Electric Power Consumption Multi-Agent Systems (RECMAS), is designed to simulate residential power consumption by multi-agent. RECMAS is composed of consumer agent, power supplier agent, and policy maker agent. It provides the policy makers with an additional tool to evaluate power price policies and public education campaigns in different scenarios. Through an influenced diffusion mechanism, RECMAS can simulate the factors affecting power consumption, and the ones associated with public education campaigns. The application of the method for simulating residential power consumption in China is presented.

Keywords: Residential power consumption; Multi-agent systems (MAS); Power price policy; Saving electricity; Social influence model.

1 Introduction

With rapid development of economy and perceptible improvement of the living standard of people, residential power demand is growing fast, which creates a strong requirement to develop power resources management schemes. The traditional methods emphasize increasing power supplies, but they do not consider saving electricity, which results in resources waste and pollution problem. However, under the energy shortage condition, enhancing the saving electricity consciousness of people, changing the habits of using-electricity, and improving using-electricity efficiency should be taken into account. Policy makers have the requirement to combine "structural" and "nonstructural" methods by considering the potential saving electricity ability of the consumers [1,2], the increasing population, and pollution. This process includes the analysis on social education, power price policy, compensating method, etc. However, it is difficult to evaluate the influence of those approaches on consumer behaviors. On one hand, the consumers have different beliefs, habits, skills and knowledge related to the

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 919–929, 2007.

environment. On the other hand, they interact with their friends, fellows, acquaintances, neighbors, etc. The problem analyzed above has brought a strong motivation to propose a solution. Because the similarity between the consumers and the way agents can be implemented, we adopt a multi-agent social simulation to evaluate the impact of policies in this paper. A hybrid social model for evaluating the impact of saving electricity policies is offered, and a platform, termed Residential Electric Power Consumption Multi-Agent Systems (RECMAS), is design to simulate the residential society in power demand-supply chain. The model benefits from previous works: studies based on econometric model [3] and successful apply of agent-based social simulation in water management [4]. RECMAS modifies the traditional econometric models by designing a social simulation layer to capture social responsiveness on saving electricity policies and account for social education strategies. RECMAS enables the user to explore the effects of the policies on total residential power demand, and it facilitates the design, creation, modification, and simulation of different scenarios.

The rest of the paper is organized as follows. Section 2 provides some facts about multi-agent social simulation. Section 3 introduces the model of residential power demand emphasizing the social environment of consumers. Section 4 describes the simulation in detail including its architecture, procedure and implementation. Section 5 gives the experiments by running a variety of scenarios. Finally, Section 6 offers conclusion remarks.

2 Multi-agent Social Simulations

Multi-agent systems establish a major research domain in artificial intelligence [5], which was focused on the resolution of problems by a society of agents. This field is increasingly characterized by the study, design and implementation of societies of artificial agents, on the extensive use of computational modeling for real-world applications and social simulations. Such a method to study complex systems is fast growing up in a wide range of scientific fields, for example, society economy [6], residential using-water management [4], power market simulation [7], and so on. It is due, for the most part, to its ability to deal with different models of individuals, ranging from simple entities to more complex ones. The distribution in several agents is also necessary because these problems may be complex or too large to be solved by a single agent, or even, they need knowledge about several fields. Generally, in a multi-agent social simulation, the society is composed of a number of agents that are able to interact with each other and the environment, and they differ from each other in their habits, skills and their knowledge about the environment.

3 Hybrid Social Model Based on Multi-agent

3.1 Residential Power Demand-Supply Chain

The residential power demand-supply chain involves three main actors. The first is the power utility company, who offer power to an area; the second is the area

residents, who consume power and undertake its costs; and the third is grid company. In China, current power retail market is not open, and the power price is enacted by the government. Ignored the grid influence, the demand-supply chain includes three main actors, i.e., power utility company, power consumers and policy maker. The government departments enact the price, while the company supplies residents with power.

Residential power demand is influenced by many factors which were summarized in the literature based on questionnaire surveys or statistic data. Influence factors of residential power consumption include family incomes, housing condition, household appliances, area and weather, promotion of saving electricity technologies, adoption of pricing strategies to discourage inefficient use of electricity, public social education, and financial incentive programs [1,2,8].

3.2 Residential Power Consumption Forecast Model

For estimating power demand, a variety of methods and econometric models have been used based on the nature and availability of data. Ref. [9] summarized four types of parametric econometric models of energy demand, whose functional forms include linear functional form, log-linear form, translog model, and almost ideal demand systems (AIDS) form. Ref. [3] presented urban residential power demand forecast model by using OLS method. Power demand estimation is usually formed as a generic model of form $D = f(P, H)$ which relates power demand D to some price measures P and family attribute H.

Econometric model can reflect the underlying relationships between the consumption of various power resources and explanatory variables such as power prices, weather variables, income, and other factors. However, it does not represent function of public social educations which are tools for educating and informing consumers on how to modify their habits of using-electricity and enhance using-electricity efficiency [1]. It is usually difficult to evaluate the results of such public social educations. On one hand, public social educations have a direct impact on consumers who participate in them; on the other hand, there is an indirect impact realized by participates, who propagate the ideas of saving electricity to their friends, fellows, acquaintances, neighbors, and so on. We can use multi-agent social simulation to simulate the results of public social educations. With influence of public social educations, the hybrid social model is formed as $D = f(P, H, S)$, which relates power demand D to some price measures P, family attribute H, and social attribute S. The power demand is defined as

$$C(i,t) = a + bP(i,t) + cH(i,t) + dS(i,t) + \varepsilon(i,t), \tag{1}$$

where $C(i,t)$ is the power consumption for consumer agent i at time t; $P(i,t)$ is the vector of price variables; $H(i,t)$ is a vector of consumer attribute variables (incomes, household appliances, habits of consumer, etc.); $S(i,t)$ is the vector of social attribute variables; $\varepsilon(i,t)$ is the error term; a, b, c, and d are coefficients to be estimated.

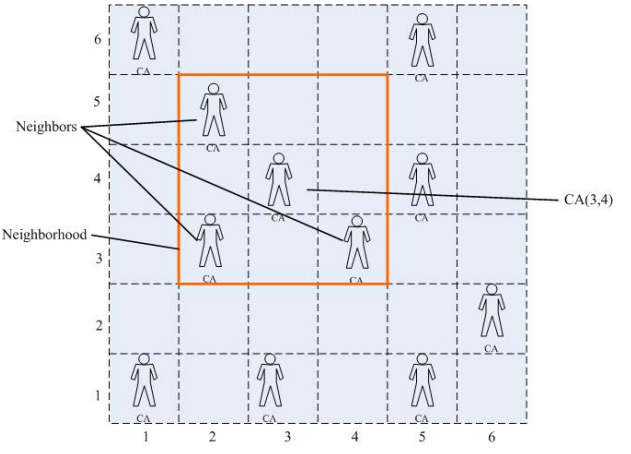


Fig. 1. Artificial residential society

3.3 Artificial Residents Society

In the power demand-supply chain, residents live in a society and interact with each other. To simulate their interaction, we begin with the formalization of the artificial residential society. We use a square social grid, as shown in Fig. 1, to simulate the society of residents. The society is composed of a set of consumer agents (CA), whose communication represents the social relationship among them. Each CA is situated on a square social grid and determined by its position on the grid. So a single CA is identified as CA(x, y), where x, y are its coordinates on the grid. Social interaction between CAs is limited to a neighborhood, defined as its neighbor environment. In Fig. 1, the 2-D grid of a side is equal to 6, the neighborhood scope is limited to 1, and the neighborhood area of CA(3, 4) is marked out by the rectangular frame. The social influence model is realized in the neighborhood area, so CA(3, 4) power consumption is affected only by its three neighbors, CA(2, 3), CA(2, 5), and CA(4, 3). So the social attribute variables can be calculated by the following equation [4]:

$$S(i,t) = f(sw_1, sw_2, \ldots, sw_n), \tag{2}$$

where $S(i,t)$ is the social variables of CAi at time t; sw_j is the social weights that CAi receives from its neighbor j, $j = 1, \ldots, n$; n is the number of neighbors of CAi; f is a diffraction function adjusting the sum of social weights, and it represents a consumer's ability to comprehend signals of saving electricity. f is chosen as S function based on people's cognitive competence, i.e.,

$$f(x) = \frac{1}{1 + e^{-(x-5)}}, \ x \in [1, 10]. \tag{3}$$

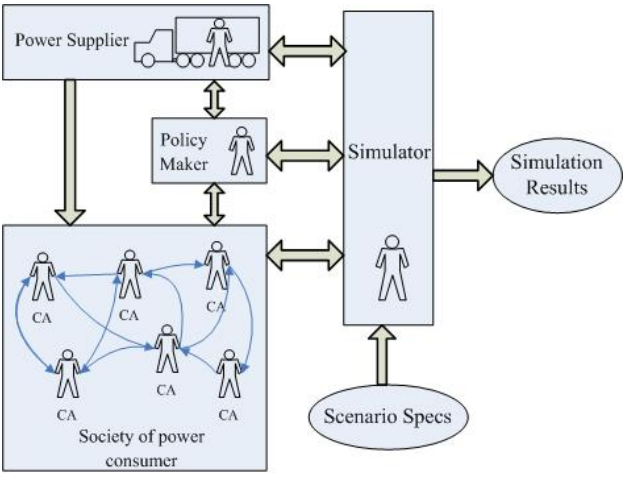


Fig. 2. Architecture of RECMAS simulation platform

4 Simulation of Residential Power Consumption

There are three classes of agents in the residential power consumption demand-supply chain: power supplier, power consumer, and policy maker. In simulation model, all participators are regarded as agents with attribute.

4.1 RECMAS Architecture

The architecture of the simulation platform RECMAS based on multi-agent, as shown in Fig. 2, includes three kinds of agents: policy maker agent (PMA), consumer agent (CA), and power supplier agent (PSA). PMA is responsible for enacting the power price in each period. CA plays the major role in the simulation, using the hybrid social model to estimate their personal demands. PSA is responsible for collecting CA's demands and calculates the total consumption, and it can give some suggestions to PMA according to its own accounting.

4.2 Agent Role

This paper designs every agent function and state in simulation platform by Gaia method [10]. It describes macroscopical and microcosmic agent role partition and function definition in agent simulation system from sociology and histology angle. It regards agent simulation system as an organization or society, and facilitates the design, creation, modification, and execution of the MAS, and defines agent role, purview, responsibility, agreement, activity and security. Anyone's role can be regarded as an entity function abstract description. In hybrid society model, we think over residential power consumption problem in one area. There is only one PSA in the area, whose main mission is offering power according to

residential demand and advising power price to PMA according to their benefit, and role schema is shown in Table 1. For consumers, CA has two roles: one is power consumer that indicates its function; the other is consumer's neighbor that indicates its society character. Its roles are shown in Table 2 and Table 3, respectively. Each CA uses the hybrid society model to estimate its personal consumption and reports its demands to PSA. Each CA communicates with its neighbors to propagandize saving electricity policy and power price policy, its influence is reflected by society weight, and the influence mechanism is shown in Section 3.3. PMA denotes policy maker, and enacts the power price according to the total power supply, demand relationship, and society environment, whose role schema is shown in Table 4.

4.3 Simulation Procedure

In RECMAS, all agents interact with each other and implement the overall simulation procedure.

Step 1. Scenario input and initialization. User specifies the scenario to be simulated, including the size of the group, CAs initial state, neighborhood limit, iteration interval time, and maximal iteration.
Step 2. PMA enacts the price and informs PSA and CAs the price.
Step 3. CA receives price inform, and communicates with its neighbors according to the social influence model.
Step 4. CA uses the hybrid social model to estimate its personal consumption.
Step 5. CA reports its demand to PSA.
Step 6. PSA collects all residents' demand and calculates the total demand and the total costs, and reports the results to PMA.
Step 7. PMA adjusts the price if needed, turn to Step 2.
Step 8. When the iteration is over, the simulation results are presented.

Table 1. The power supplier role schema

Role	Power Supplier Agent
Description	Simulating power suppliers behavior.
Activities	WaitStartStep, QueryCustomer, CalculateStepTotalConsumption
Protocols	SubmitTotalComsumption, SendPriceAdvice, GetCustomerConsumption
Permissions	Read: PowerPrice, PersonalConsumption, Write: StepTotalConsumption, Send: PriceAdvice
Responsibilities	
Liveness	PSA=(WaitStartStep · GetCustomerConsumption · CalculateStepTotalConsumption · SubmitTotalComsumption · SendPriceAdvie)$^{+}$ [a]
Safety	True

Table 2. The power consumer role schema

Role	Power Consumer Agent
Description	Simulating power consumers behavior.
Activities	WaitStartStep, ConsumePower
Protocols	SendPersonalComsumption, ContactNeighbor, ReceivePrice
Permissions	Read: PowerPrice, NeighborsList, PersonalParameters, LastStepConsumption, StepID Write: StepPersonalConsumption
Responsibilities	
Liveness	PSA=(WaitStartStep · ReceivePrice · ContactNeighbor∗ · ConsumePower · SendPersonalComsumption)$^{+}$[a]
Safety	PersonalPowerConsumption>0

Table 3. The neighbor role schema

Role	Neighbor
Description	Simulating consumers neighbor behavior.
Activities	CalculateWeights
Protocols	ReplyNeighbor
Permissions	Read: NeighborsList, SocialWeightParameters, StepID Write: SocialWeights
Responsibilities	
Liveness	PSA=(CalculateWeights · ReplyNeighbor)*[a]
Safety	Neighbor [neighborList]

Table 4. The policy maker role schema

Role	Policy Maker Agent
Description	Simulating policy maker behavior.
Activities	DecidePolicy
Protocols	SendPolicy
Permissions	Read: TotalPowerDemand, Changes: Policy
Responsibilities	
Liveness	PSA=(DedidePolicy · SendPolicy)$^{+}$[a]
Safety	True

[a] $x \cdot y$ notes x followed by y, x^+ notes x occurs 1 or more times, x^* notes x occurs 0 or more times, $[x]$ notes that x is optional.

5 Experiment Analysis

In this section, we first prove the validation of the model. Next, we forecast urban residential annual average power consumption in 2006–2010 and analyze the influence of power price policy, residential incomes, and public social education factors on residential power consumption based on six scenarios.

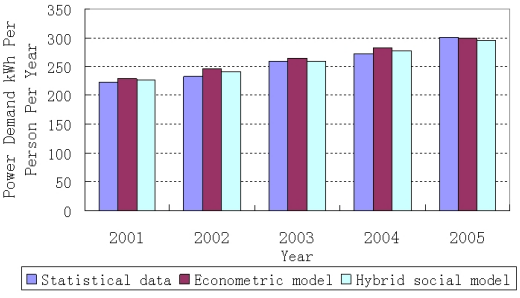


Fig. 3. Comparison of econometric model and hybrid social model

5.1 Calibration and Validation of the Model

In order to prove the validation of the model, using data in [3], we forecast urban residential power demand in the period of 2001–2005 in China. In simulation environment, we set 10×10 resident society spaces and 4 classes 80 to 100 units randomly. The results are shown in Fig. 3. The hybrid social model is used to analyze the influence of policy on the residential power demand from 2006 to 2010 in China.

5.2 Simulation Experiment Scenarios

The platform was used to simulate six different scenarios, which are as follows:

- Scenario 1: Power price is increased by 5 percent, without considering the public social educations.
- Scenario 2: Power price is increased by 7.5 percent, without considering the public social educations.
- Scenario 3: Power price is adjusted to the real price, considering the public social educations, with the implementation of education or other information policy of medium scale.
- Scenario 4: Power price is increased by 5 percent, considering the public social educations, with the implementation of education or other information policy of medium scale.
- Scenario 5: Power price is increased by 7.5 percent, considering the public social educations, with the implementation of education or other information policy of major scale.
- Scenario 6: Consumers' income remain unchanged every year.

5.3 Consumer Types

Residential energy consumption value mode was analyzed in [11] by society investigate questionnaire, and eight kinds of consumption conceptions were summarized; four types of consumers were analyzed in [4]. This paper combines the results in [4] and [11], consumer agents are clustered in four types, with respect to their abilities to promote and comprehend saving electricity signals.

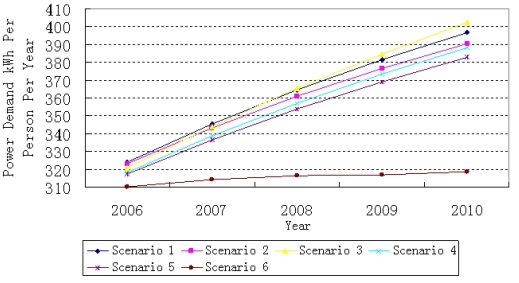


Fig. 4. Simulation results

- Type A: Opinion leaders, 10 percent, are supposed to be environmental aware; their ability to further lower their power consumption is generally limited.
- Type B: Socially apathetic, 20 percent, neither promote nor comprehend saving electricity signals and have a negative attitude about conservation.
- Type C: Opinion seekers, 30 percent, are supposed to be socially sensitive and act as opinion followers. They can be easily influenced by families through their social relations with opinion leaders.
- Type D: Opinion receivers, 40 percent, are the average consumers. Their attitude is passive, as they need considerable pressure by their contacts to start to change their habits of using-electricity.

5.4 Simulation Analysis Results

Simulation platform was fulfilled by JAVA based on MAGE [12]. Each simulation period was supposed to be one year. The quantitative estimations obtained for the period from 2006 to 2010 are shown in Fig. 4 in the form of personal demand (kWh/year).

A simulation for Scenario 6 was run under the assumption of the consumers' income being constant every year, with slow increase in power demand. Compared with other scenarios, the experiment results show that the growth of income is the main reason that leads to the power demand growth in China. The comparisons between Scenario 1 and Scenario 4, Scenario 2 and Scenario 5 indicate that public social education and information policy can promote consumers' consciousness of saving electricity and make more efficient use of electricity. Thus, through public social education, about 2 percent power consumption can be saved per capita every year. In addition, the suitable power price policy is an available measure to transfer information of saving electricity. The comparison between Scenario 1 and Scenario 3 shows that if the power price is increased by 5 percent, about from 2 to 3 percent power consumption can be saved per capita every year. The comparison between Scenario 2 and Scenario 3 shows that if the power price is increased by 7.5 percent, about from 3 to 4 percent power consumption can be saved per capita every year.

6 Conclusion

In this paper, we proposed a hybrid society model and RECMAS for analyzing residential power consumption. The hybrid social model extends traditional models by integrated social influence model. RECMAS modifies the traditional econometric models by designing a social simulation layer to capture social responsiveness on saving electricity policies and account for social education strategies simulation. RECMAS is designed based on multi-agent systems, where different policy scenarios can be adopted to get the influence of power price, residential incomes and public social educations on residential power consumption. From the simulation results, we obtain that the growth of income is a main reason that leads to the power demand growth in China; and public social education and suitable power price policy can effectively promote consumers' consciousness of saving electricity and make more efficient use of electricity.

The superiority of RECMAS is that it supports communication and influence between consumers in society. Every consumer communicates with its neighbors to propagandize saving electricity policy and power price policy, change their habits of using-electricity, and improve using-electricity efficiency according to their society characters. RECMAS is a real residential society simulation platform and can offer a policy simulation environment for decision-maker to reduce the cost of actualizing policy.

References

1. Vringer, K., Aalbers, T., Blok, K.: Household Energy Requirement and Value Patterns. Energy Policy 35, 553–566 (2007)
2. Dintchev, O.D., Calmeyer, J.E., Delport, G.J.: Efficient and Sustainable Usage of Electricity in South Africa: the Role of the Tertiary Education Institutions. Engineering Science and Education Journal 9, 53–59 (2000)
3. Song, W., Gu, A.L., Wu, Z.X.: Urban Residential Electricity Consumption Forecast Model. Electric Power 39, 66–70 (2006)
4. Athanasiadis, I.N., Mentes, A.K., Mitkas, P.A., Mylopoulos, Y.A.: A Hybrid Agent-Based Model for Residential Water Demand. Simulation 81, 175–187 (2005)
5. Weiss, G.: Multi-Agent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, Cambridge (1999)
6. Lopez, A., Hernndez, C., Pajares, J.: Towards a New Experimental Socio-Economics: Complex Behavior in Bargaining. Journal of Socio-Economics 31, 423–429 (2002)
7. Yuan, J.H., Ding, W., Hu, Z.G.: Acritical Study of Agent-Based Computational Economics and its Application in Research of Electricity Market Theory. Power System Technology 29, 47–51 (2005)
8. Moholkar, A., Klinkhachorn, P., Feliachi, A.: Effects of Dynamic Pricing on Residential Electricity Bill. Power Systems Conference and Exposition, IEEE PES 2, 1030–1035 (2004)
9. Zarnikau, J.: Functional Forms in Energy Demand Modeling. Energy Economics 25, 603–613 (2003)

10. Wooldridge, M., Jenning, N.R., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. Autonomous Agents and Multi-Agent Systems 3, 285–312 (2000)
11. Hessing-Couvret, E., Reuling, A.: The WIN-Model, Value Systems in the Netherlands. TNS-NIPO, Amsterdam (2002)
12. Shi, Z.Z., Zhang, H.J., Dong, M.K.: MAGE: Multi-Agent Environment. In: ICC-NMC 2003 (2003)

On Intelligent Interface Agents for Human Based Computation*

F. Aznar, M. Sempere, M. Pujol, and R. Rizo

Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
{fidel,mireia,mar,rizo}@dccia.ua.es

Abstract. In this paper a new type of interface agent will be presented. This agent is oriented to model systems for human based computation. This kind of computation, that we consider a logical extension of intelligent agent paradigm, emerges as valid approach for the resolution of complex problems.

Firstly an study of the state of the art of interface agents will be review. Next, human based computation will be defined and we will see how is necessary to extend the current typology of interface agents to model this new kind of computation. In addition, a new type of interface agent, oriented to model this type of computational system, will be presented. Finally, two of the most representative applications of human based computation will be specified using this new typology.

1 Introduction

Knowledge and communications can be considered as two of the most important pillars of our society. Whereas the information of digital systems increases (such as products, services, people, maps, ...), the computing technologies we have develop become, paradoxically, both the gateways to all kinds of resources and the barriers to access them. Therefore, computers become very important in improving our lives and are essential not only to the relatively few of us who have the necessary skills to access to resources, but to everyone [1], [2]. More and more, we must use very specific interfaces to carry out some communication tasks with computers and other electronic devices.

Thinking of this problem, as early as in the decade of the 90, Kay [3] and other authors, being based on the paradigm of intelligent agents, emphasized the importance of an interface agent, that was going to improve the computation as we know it today. An interface agent would allow us to advance from the direct system manipulation to simpler interaction based on agents. This way, the elimination of the necessity of controlling all the interface details allows people to have more time to do another things (people will be able to obtain others goals, that in another way would require an expert).

Instead of establishing a direct interaction with commands or a direct manipulation of the interface, the user is introduced in a cooperative process where the

* This work has been financed by Generalitat Valenciana Project ARVIV/2007/071.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 930–939, 2007.

human agent and the interface agent start a communication, monitoring a set of events and carrying out a group of tasks. An analogy between an interface agent and a personal assistant can be drawn, because this agent is collaborating with the user in the same work environment [4]. If this cooperative process between the user and the interface agent is established in a closer way, even symbiotic, we find a new computational paradigm: human based computation.

Human based computation can be defined as a technique that performs computational problems requesting the resolution of certain steps of the problem to humans. This approach is different from traditional computation, where a human employs a computer to solve a problem. In this computation, a human provides a formalized problem description to a computer and receives a solution to interpret it. In human based computation, the roles are often reversed: the computer asks a human (or several) to solve a problem, collecting, interpreting and integrating all the solutions.

The importance of this computation, that makes that companies as google (`www.google.com`) includes them in search engines, lies in the use of the innate ability of humans to detect patterns in order to help computers (that are less suitable to this task, although computers are better than humans in other tasks, such as detecting randomness in a process, ...) [5].

In this paper, we will review the state of the art of current interface agents that learn from the user, analysing their characteristics, advantages and disadvantages. Then, a new type of interface agent able to perform human based computation tasks (HC), will be specified. Therefore, we will show both the characteristics that define a human based computation process and the roles and actors that must be present in this type of computation. Although some applications based on HC exist, we have not found any specification or formalization based on intelligent agents, being human based computation, from our point of view, a logical extension of interface agents. Some of the most representative applications based on human based computation will be analysed from the point of view of interface agents. Finally conclusions and future work will be drawn.

2 Intelligent Interface Agent Systems

An interface agent can be defined as an intelligent system which assists users with daily computer based tasks [6]. This agent allows the user to delegate difficult tasks with a main purpose, to reduce the workload of users by creating personalized agents to which personal work can be delegated.

From the point of view of agents that learn from the user, three types of interface agents exist: user programmed agents, where the user must program the agent; agents that receive user feedback, where the interface agent is endowed with an extensive domain model and a user model, and agents that monitor user behaviour, which are provided with the ability to program theirself, for example, acquiring the knowledge that they need to assist the user [4].

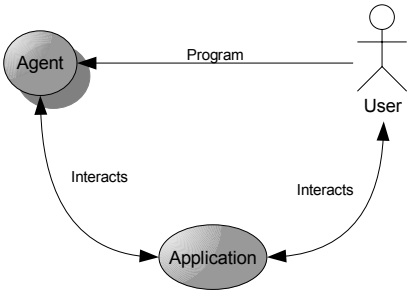


Fig. 1. Interaction between actors in an user agent programmed

When designing this type of agents it is necessary to consider two main criteria: the competence of the agents, that is, it should be specified how the agent acquires the knowledge that needs to decide when and how to help the user; and their trust, we should know how we can guarantee that the user feels comfortable delegating the tasks to the agent.

The user programmed interface agents are the simplest to implement but they are not always the most appropriate, because their operation not always denotes intelligence. In this type of agents, end users are who directly provide the rules and the criteria of operation of the agent. Their main problem is that their operation depends directly on the abilities of the user to program the agent.

The operation of this type of agents is summarized in the figure 1 where the main flows of information and the actors that are part of the environment are shown. As can be observed, an agent is programmed by the user to help him with the interaction with the application.

These agents do not satisfy the competence criterion since they require a lot of knowledge and user effort. In addition, the user has to recognize the opportunity to use the agent, to take the initiative to create it, to provide the necessary knowledge as well as to maintain it consistent in time. Regards trust criterion, this type of agents are not problematic because the user is who program them.

Agents that receive user feedback. These type of agents are usually created using knowledge based systems. These agents could receive explicit feedback or only an initial training. Their development is quite complex but they usually show intelligent behaviours.

In figure 2 can be observed the operation of this type of agents. A knowledge engineer must program the agent in a way that it accepts the knowledge of the user. The agent will use this knowledge to interact with the application and to improve the interaction of the user with the application.

Regards the competence, these agents need a lot of knowledge to be embed into the agent. Sometimes, this agent cannot be reused in other applications. On one hand, in the case that the agent only receives an initial training, the knowledge of the agent is prefixed so cannot be adapted to habits or preferences of the user. On the other hand, if the agent is programmed by an external person

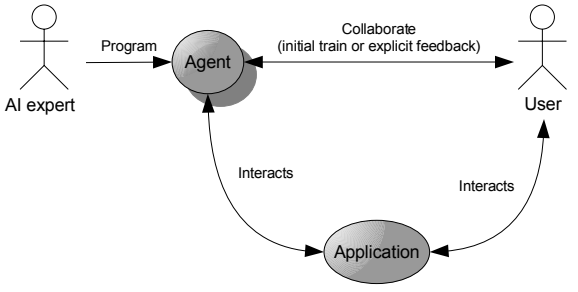


Fig. 2. Interaction between actors in an agent that receive user feedback

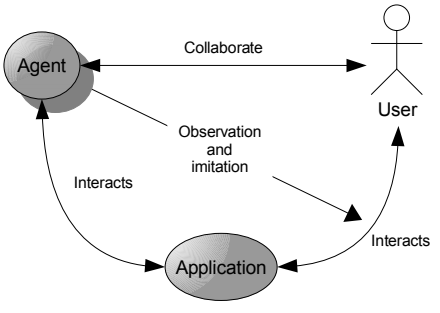


Fig. 3. Interaction between actors in an agent that monitor the user

the trust is reduced because, for example, the agent could have programmed a model that the user does not know.

Agents that monitor user behaviour extract patterns from the actions of the user or even of other users. The complexity of developing these agents is middle and its operation should be correct in specific areas. An agent of this type needs some knowledge of the environment to learn a *correct* behaviour from the user or other users.

The operation of this type of agents can be observed in the figure 3. Both the agent and the user interacts with the application. Nevertheless the fundamental difference with previous approaches is that the agent monitors the interaction between the user and the application to learn from it. This type of agents acquires its competences from four sources of information:

- Monitoring the user while the user is interacting with the application
- With direct or indirect feedback with the user
- From examples provided by the user
- Asking to other agents that assist other users and have the same task

Agents that monitor user behaviour will be more and more competent as they learn the user preferences. Regarding the trust, these agents will develop

their abilities gradually. The agent will also be able to give explanations of its reasoning and behaviour in a language close to the user.

3 Interface Agents for Human Based Computation

As we commented above, human based computation can be defined as a technique that performs computational problems requesting the resolution of certain steps of the problem to humans. This technique is being used for years and now it is very important due to the growth and development of information networks. Next, we will present a new type of interface agents oriented to perform this type of computation.

If we base on the basic characteristics of a flexible system, it is easy to realize that an intelligent agent perfectly adapts to this kind of computation. On one hand, it must be a social agent, that is, it must interact in an appropriate way with other agents and humans so that it must be able to complete its problem helping others with their tasks. On the other hand, it must be pro-active and therefore, it must act in two ways: answer to the environment and be able to look for the opportunity and to take the initiative when it considers appropriate.

In the classification of interface agents presented in the previous section the agent that is adapted better to the human based computation is the agent that monitors user behaviour. Nevertheless there are some characteristics that are not captured in this type of agents and are necessary for a human based system:

New sources of information. In agents that monitor user behaviour the information is obtained from the interaction of the user. In human based systems, the agent can request to the user or to other agents some help to solve a computational problem.

Deeper interaction between agents and the user. A new communication more deeper than the feedback between agents and humans exist. In contrast to classic interface agents where the interface agent is considered as an assistant of a human, in this case, both human and artificial agents must collaborate in the same level to solve a certain problem.

Completely different agents can collaborate to each other to perform this type of computation. Collaborative systems of interface agents are usually based on the exchange of information between agents with the same characteristics that interact with different users [6]. In human based computation can exists different agents that are able to exchange information obtained from the users to solve a certain problem.

This way, we see the necessity to specify a new type of interface agent oriented to human based computation.

3.1 Agents for Human Based Computation

An agent system for human based computation can be defined as a group of intelligent agents that are organized as is shown in figure 4. These agents must have the next characteristics:

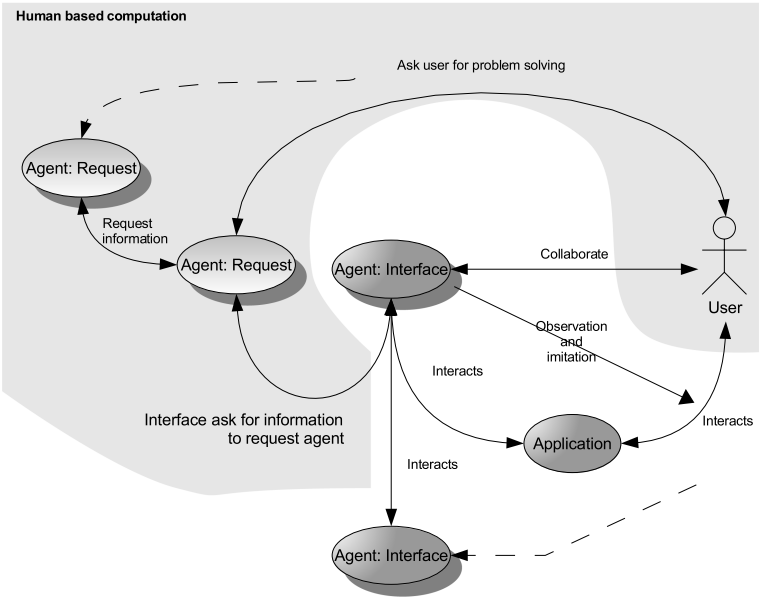


Fig. 4. Interaction between actors in an agent based on human computation

Request agents. There are some agents that request information to the user. These agents form the basic process of human based computation. They set out computational processes to human so that humans must solve them collaborating with the system. Request agents can request information to other agents with the same role, this way, an agent can obtain information from several users.

Request role is not exclusive and an interface agent can develop this role, so that, the minimum system based on human computation could have only one agent that develop the request role.

Any agent that communicates with the user can provide the request role and therefore it can provide valuable information about computational problems that it has delegated to the user. The agents that act as request could be very different from each other (in fact, the more different, more information they will obtain from the user).

Resolution of computational problems. Request agents can ask for the intervention of the user in different ways, always to solve a computational problem that they cannot deal with. It can be an explicit request or also it should be common to develop an agent for a different application (normally playful) designed specifically to obtain the data needed from the user (a lot of times the user is not conscious of this).

Interface agents. Interface agents act in the same way than the agents that monitor user behaviour but these agents can obtain information from a new source asking to request agents.

It is important to highlight that the information used by request agents is not the same that the information of other interface agents. As we commented above, interface agents have the same design for a certain application and therefore their behaviour is very similar. In contrast, request agents can be very different and therefore they can provide more information than a group of interface agents.

The fundamental difference between the roles of interface agents and request agents is that interface agents make predictions, comparing new situations with older situations and they try to obtain confidence measures of the system operation, for example, asking to other interface agents. In contrast, the main purpose of request agents is to ask for the solution of a computational problem and obtain and process the result of this problem.

Interface agents systems based on human computation, like agents that monitor user behaviour, are gradually being more competent. Moreover, the learning process is faster due to the fact that they can ask for more specific information to request agents. In the same way, an agent will develop its abilities whereas the user learns the agent operation. In this case, the agent has been designed using data obtained from human computation, so its operation is closer to the user, since the agent can even give explanations extracted directly from human reasoning.

The main advantages of the use of agents based on human computation can be summarized in the following points:

- In human computation, each agent try to solve those problems that it is able to face up to in a better way, simplifying the resolution as well as improving the obtained results.
- The results obtained by a lot of users are fused in order to obtain a general conclusion that can be used for other problems or even as a base for a general resolution model. In this paper we will not discuss the advantages of information fusion, that can be consulted in [7], [8], [9].
- Many applications of human computation take advantage of the free time of humans in order to perform useful operations, using this time of calculation that a priori it is wasted.

4 Reviewing Human Based Computation Applications

In this section two of the most common applications of human based computation will be analysed from the point of view of intelligent agents. This way we will see how these applications can be formalized using the new type of interface agents proposed.

4.1 Google Image Labeler

Several of the most important applications related to the field of human based computation arise from the team directed by Luis von Ahn. For example, Phetch

[10] is an on-line game that tries to improve the accessibility to the Web. Players have to label a set of images that will be used to provide alternative texts. This is very important to provide access to visual impaired people. Peekaboom [11] is other collaborative game where a player tries to guess the image that the opponent is hiding. The opponent only shows some parts of the image. Data obtained by this game are used to train an artificial vision system.

In this section we will model the main interactions of the application Google Image Labeler (`http://images.google.com/imagelabeler`). In this application two people must be in agreement to assign tags, labels and keywords to random images that come from Google Images. The more images you guess, the more points you get. The result will be quite good if two people were able to be in agreement, taking into account the difficulty of this problem for a computer. This application was of a project of the research group directed by Luis von Ahn and was licensed to Google Inc.

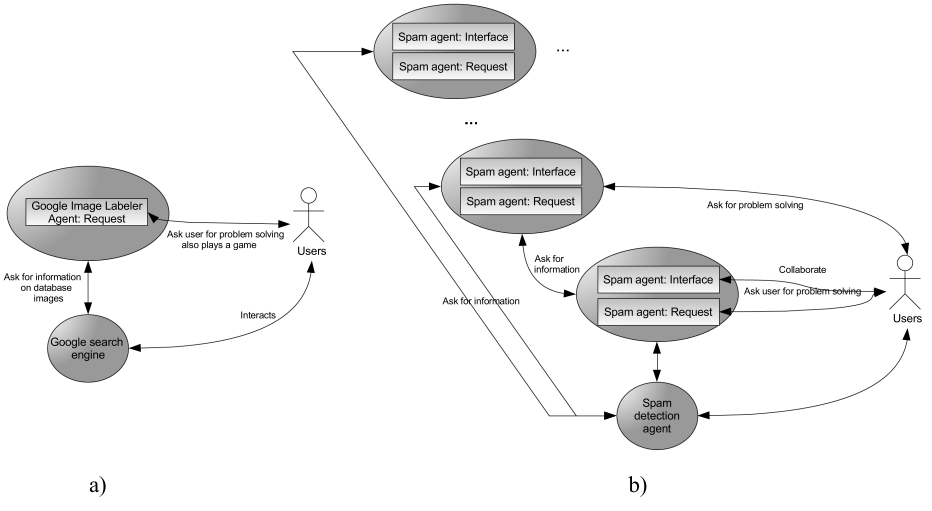


Fig. 5. a) Viewing Google image labeler as an agent system, b) Spam filter system modeled with interface agents

As can be seen in the figure 5 Google Image Labeler is a playful application that implements a request agent. This agent obtains information that will be used in google search engine. This engine could use different interface agents that interact with users, but by default any agent of this characteristic is provided so the user interacts directly with the application.

Google Image Labeler is one of the most representative applications of human based computation, that is not modelled with agent paradigm. Nevertheless, it is a usual structure of applications for human based computation that have not been designed with an agent oriented methodology. An application for human based computation must have al least one request agent interacting with the user

and with the main application, as in this case. User experience can be improve integrating an interface agent to this application. On one hand frequent questions of the users could be learned. In addition, other users that use the same searching patterns can be looked for to share results. Some of the improvements proposed can be consulted in [6].

4.2 Spam Detection

Other applications that use human based computation are spam detection applications. In Spam net web site (`http://www.cloudmark.com`), Vipul's Razor web site (`http://sourceforge.net/projects/razor`) and in [12] each mail received by the user is marked as spam with a specific button (if it is the case) by the user. This way when some people mark the mail as spam, this mail will be considered automatically spam for the rest of users.

The figure 5b shows the general interaction diagram of these applications. In this case, some request agents that obtain information from the users and from others agents are defined to provide a global knowledge. In addition, each request agent shares the role of interface agent, and must collaborate with the user. To split both roles in different agents depends on the problem to solve. Sometimes it is relatively easy to create external applications with agents that developing the request role will obtain the needed data for a certain tasks. Whenever it is possible, is a good option, since it can allow to access to a bigger number of users.

5 Conclusions

In this paper a brief state of the art of interface agents has been presented. Based on this review a new type of interface agent, that can model human based computation process, has been specified.

Moreover, human based computation process has been review, specifying its most important features, and analyzing them as a logical extension of the intelligent agents paradigm. Also we have showed two of the most important applications of this kind of computation, providing an agent based design, using the proposed type of interface agent.

We see this work as a necessary basis for elaborating interface agent systems that use human based computation. This kind of systems can use all the advantages of both, agent paradigm and human based computation. Based on the proposed model we are now developing applications oriented to computer vision and robotics for learning human behaviour patterns that could be useful for computers.

References

1. Brewer, E., Demmer, M., Du, B., Ho, M., Kam, M., Nedevschi, S., Pal, J., Patra, R., Surana, S., Fall, K.: The Case for Technology in Developing Regions. Computer 38(6), 25–38 (2005)

2. Sims, K.: Artificial Evolution for Computer Graphics. SIGGRAPH Comput. Graph. 25(4), 319–328 (1991)
3. Kay, A.: User Interface: A Personal View. Addison-Wesley, Reading (1990)
4. Maes, P.: Agents that Reduce Work and Information Overload. Comunications of the ACM 37 (1994)
5. Norving, P.: Mistakes in Experimental Design and Interpretation (2007), http://norvig.com/experiment-design.html
6. Lashkari, Y., Metral, M., Maes, P.: Collaborative Interface Agents 1 (1994), citeseer.ist.psu.edu/lashkari94collaborative.html
7. Llinas, J., Bowman, C., Rogova, G., Steinberg, A., Waltz, E., White, F.: Revisiting the Jdl Data Fusion Model II. In: Proceedings of the Seventh International Conference on Information Fusion (2004)
8. Waltz, E., Llinas, J.: Multisensor Data Fusion. Artech House (1990)
9. Brooks, R., Iyengar, S.S.: Multi-Sensor Fusion: Fundamentals and Applications. Prentice Hall, New Jersey (1998)
10. von Ahn, L., Ginosar, S., Kedia, M., Liu, R., Blum, M.: Improving Accessibility of the Web with a Computer Game. In: CHI 2006: Proceedings of the SIGCHI conference on Human Factors in computing systems, pp. 79–82. ACM Press, New York (2006)
11. von Ahn, L., Liu, R., Blum, M.: Peekaboom: A Game for Locating Objects in Images. In: CHI 2006: Proceedings of the SIGCHI conference on Human Factors in computing systems, pp. 55–64. ACM Press, New York (2006)
12. Zhou, F., Zhuang, L., Zhao, B.Y., Huang, L., Joseph, A.D., Kubiatowicz, J: Aproximate Object Location and Spam Filtering. In: ACM Middleware (2003)

Reverse Engineering an Agent-Based Hidden Markov Model for Complex Social Systems

Hung-Ching Chen, Mark Goldberg, Malik Magdon-Ismail, and William A. Wallace

Rensselaer Polytechnic Institute, Troy, NY 12180, USA
{chenh3, goldberg, magdon}@cs.rpi.edu, wallaw@rpi.edu

Abstract. The power of social values that helps to shape or formulate our behavior patterns is not only inevitable, but also how we have surreptitiously responded to the hidden curriculum that derives from such social values in our decision making can be just as significant. Through a machine learning approach, we are able to discover the agent dynamics that drives the evolution of the social groups in a community. By doing so, we set up the problem by introducing an agent-based hidden Markov model, in which the acts of an agent are determined by *micro-laws* with unknown parameters. To solve the problem, we develop a multistage learning process for determining the *micro-laws* of a community based on observed set of communications between actors without the semantic contents. We present the results of extensive experiments on synthetic data as well as some results on real communities, *e.g.*, Enron email and movie newsgroups.

1 Introduction

Each day, individuals from all parts of the social ramifications respond and react toward the values they perceive from the world. In the past decades, high tech has been integrated aggressively into our daily life. The rapid exchanges of communication between individuals have gone from surfing online for information to providing information, building individual Space / Blog as well as getting connected through various instant messaging communities. It is apparent that online communities have become one of the influential medium to the journey of social evolution. Yet, regardless of the impact of the online communities; the role of social value continue to play an imperative factor on the dynamics of the online communities as it has been for the offline communities rapid growth, sudden emergence or hastily dissipated due to changes of demands, needs, and values of the existing society. Therefore, it is essential to acquire ranges of more comprehensive and objective social factors that might have propelled the evolution of the society.

A social group is a collection of agents, or actors who share some common context [1]. The dynamics of the social groups are governed by the actor dynamics actors join groups and leave groups. An actors actions are governed by collective values that are direct or indirect results of the social context: personal attributes, the actions of other actors, and the social structure in the community. In summary, any reasonable model for an agent based evolving community must necessarily involve complex interactions between actors attributes and the social group structure itself. Therefore, the explosion of online communities provides an ideal pasture on which to groom and test social science theories, in particular the most natural question is: *what are the micro-laws [2]*

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 940–949, 2007.

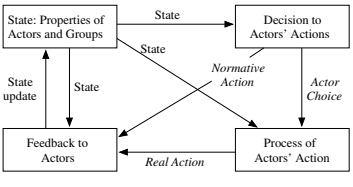
which govern a particular society? Furthermore, an efficient approach to answering this question on a given community also yields a powerful tool for a sociologist.

Due to the growing popularity and interests of social network analysis (SNA), researchers have started to use different methods to help them collect and study the structure of the social network as well as analyze the ranges / factors of social dynamics, [3,4,5]. In this paper, We uses an agent-based hidden Markov model of a social system to identify the appropriate micro-laws of a community (appropriate parameters in the model) based on the set of observable communication edges between actors without semantic contexts. Our approach uses a multistage learning process to reduce the learning complexity and also the noise in the communication data. We identify the appropriate micro-laws by solving a mixed optimization problem because our model combines discrete and continuous parameters, and to avoid the resulting combinatorial explosion, we appropriately approximate and optimize the objective within an expectation-maximization setting. To test the quality of our approximations and the feasibility of the approach, we present the results of extensive experiments on synthetic data as well as some results on real data about Enron email and movie newsgroups.

Paper Organization. Next, we briefly give an overview of the agent-based hidden Markov model in Sec. 2. Then, we present our approach to learning the group structure and evolution from the observed communications and also learning the appropriate parameters of the model in Sec. 3. We, then, give experimental results on synthetic data and real data in Sec. 4 and conclude in Sec. 5.

2 Overview of Agent-Based Hidden Markov Model

We give a overview of the probabilistic evolving social group model ViSAGE, details of which can be found in [6], and the communication model. The foundation of ViSAGE is a Markov process, and the figure on the right hand side shows the process for each time step. There are actors, groups, the *state* of the society which is defined by properties of the actors and groups. There are three kinds of actions – *join a group*, *leave a group*, and *do nothing*. Based on the current *state* of the society, each actor decides which action she most likely wants to execute, which is known as the *Normative Action*. Nonetheless, under the influence of the present communities, actions of actors are affected. After being influenced, each actor eventually performs the *Real Action*. Depending on the feedbacks from actors' *Normative Action* and *Real Action*, properties of actors and groups are updated accordingly. We also develop a communication model to produce communication networks based on the *state* of the society; in which we only consider about the communication edges between actors without the semantics of the communications. Combining ViSAGE with the communication model, the whole process represents a hidden Markov model (HMM).



2.1 State of the Society

Many parameters govern how actors decide whether to join or leave a group, or do nothing, and also which group the actor desires to join or leave; *i.e.*, parameters such

as the group memberships, how long each actor has been in a group, the ranks of the actors and the amount of resources available to an actor (in excess of what is currently being used to maintain the actor's current group memberships). Thus, the state of the society at time t can be defined as

$$S_t = \{type_i, \boldsymbol{r}_i{}^t, R_E^{i\ t}, \{G_l^{i\,t}\}_l\}_{i=1}^N, \tag{1}$$

where includes N actors, $i = 1 \cdots N$, a set of groups $\{G_l^{i\,t}\}_l$ actor i joined at time t, and the properties of actors and groups describing as follow:

Type. Each actor has a private set of attributes, which we refer to as its *type*. In the model, an actor's type simply controls the actor's group size preferences and her "ambition" (how quickly her rank increases in a group). There are 3 kinds of type in the model; *Leader* who prefers small groups and is the most ambitious, *Follower* preferring large groups and the least ambitious, and *Socialite* in the middle.

Rank. Each actor has a rank in each group to present the actor's position in the group. As actors spend more time in a group, their position in the group changes. There is a tendency for more senior members to have a higher position than junior members. $\boldsymbol{r}_i{}^t$ is the set of ranks of actor i in all groups at time t.

Qualification. Each actor has a qualification to represent an actor's prestige. It is determined as the average rank of the actor among all the groups, and the rank is weighted to give a stronger weight to ranks from larger groups. The qualification of an actor is used for an actor to determine which group she more likely join or leave. Similarly, each group has its $qualification$ defined as the average qualification of actors currently participating in the group. The higher a group's qualification, the more attractive it will appear to other actors looking for a group to join.

Resources. We use R_E^i as the available resources for actor i , and R_E^i depends on how many resources an actor needs to maintain a membership in a group. In addition, the actors' ranks and the number of groups the actor is in influences how many resources an actor needs to maintain a membership in a group. And R_E^i also influences what kind of action an actor can complete at next time step.

2.2 State Transitions

At each time step, every actor needs to decide on leaving one group, joining one group, or remaining in the same groups. The decision depends on an actor's available resources (R_E^i). The actor will tend to join another group when she has more R_E^i; otherwise, the actor will tend to leave a group in order to lessen the cost needed. We call this action as $Normative$ action. Ideally, the actor would always choose to perform the $Normative$ action, since this creates a state of stability. However, we assume that the actors sometimes make non-rational decisions, regardless of the amount of available resources they have. An actor chooses an action she is going to perform based on a stochastic process. After an actor has chosen which action she would like to perform, she needs to decide which group to join or leave. The actor takes into account the size and qualification of the group during decision making. The group can accept or reject the actor's application based a stochastic process, which is related to the group's qualification and the actor's qualification.

The final step of the process at each time step is to update the properties of actors and groups. To update properties of actors and groups is based on all actors' $Normative$ actions and real actions and the society reward/penalty parameters θ_{reward}. The reward/penalty parameters θ_{reward} determine how to update an actor's resources, and it is summarized heuristically by $Reward\left(action, R_E^i, \theta_{action}, \theta_{reward}\right)$, where θ_{action} indicates some parameters related to actors' actions.

2.3 Communications

We have developed a social networks model to produce the communication links between actors without considering the semantics of the messages. The basic idea is that the more joined groups two actors have in common, the higher probability these two actors should communicate with each other; however, if two actors have no any joined group in common, they still have a chance to communicate with each other. A more general model also consider actors' friends; if two actors are not in a same group but they have a common friend, then there is another probability for this kind of communication. We can also consider how many levels of the friendship, *e.g.*, friend's friends in common or friend's friends' friends in common, etc.

3 Learning Process

The common learning algorithms for solving the problems in a hidden Markov model are like *forward-backard algorithm* [7], *Viterbi algorithm* [8], and *Baum-Welch algorithm* [9]. The complexities of these three algorithms are the same, $O(TM^2)$, where T is the total time steps, and M is the number of states. In our model, if there are N actors and K groups in a society, then, in each time step, there are $(2^{(NK)}/K!)$ possible actors' combinations for group structure, the term $\{\{G_l^{i^t}\}_l\}_{i=1}^N$ in (1). If we have data for T time steps, the complexity of using the above algorithms is $\Omega(T \times \frac{2^{NK}}{K!}) \approx \Omega(T \times 2^{K(N-logK)})$, which is exponential computation time and is very time consuming. Therefore, we develop a multistage learning process. In the first stage, we find the group structures at each time step based on the communication networks, and then we discover the group evolution using the group structures. In the last stage, we learn from the group evolution to identify the appreciate parameters in ViSAGE.

3.1 Learning from Communications

The challenge with real data is that the groups structure and their evolution are not known, especially in online communities. Instead, one observes the communication dynamics. However, the communication dynamics are indicative of the group dynamics, since a pair of actors who are in many groups together are likely to communicate often. One could place one more probabilistic layer on the model linking the group structure to the communications, however, the state space for this hidden Markov model would be prohibitive. We thus opt for a simpler approach. The first step in learning is to use the communication dynamics to construct the set of groups.

In our communication model, for instance, let P_g be the probability that two actors in each same joined group would like to communicate, and P_b be the probability that two actors having no any same joined group would like to communicate. Let i, j refer to actors, and x_{ij} be a boolean value presenting the communication between actor i and j. Then the problem can be define as maximizing

$$Prob = \prod_{i,j} P_e(i,j)^{x_{ij}}(1-P_e)^{(1-x_{ij})}, \tag{2}$$

where $P_e(i,j) = \begin{cases} P_b & \text{, if } i,j \notin \text{ same group.} \\ 1-(1-P_g)^{k_{ij}} & \text{, if } i,j \text{ have } k_{ij} \text{ groups in common, and } k_{ij} > 0. \end{cases}$

We need to find P_b, P_g, and x_{ij} for all i, j, but any reasonable formulation of this problem is NP-hard, and so we need some efficient heuristic for finding the clusters in a graph that correspond to the social groups. In particular, the clusters should be allowed to overlap, as is natural for social groups. This excludes most of the traditional clustering algorithms, which partition the graph. We use the algorithms developed by Baumes et al. [4], which efficiently find overlapping communities in a communication graph. We consider time periods $\tau_1, \tau_2, \ldots, \tau_{T+1}$ and the corresponding communication graphs $G_{\tau_1}, \ldots, G_{\tau_2}$. The time periods need not be disjoint, and infact choosing them to overlap is preferable since there is considerable noise in the communications – aggregation, together with ovelap smoothens the time series of communication graphs. Given a single graph G_{τ_t}, the algorithms in [4] output a set of overlapping clusters, $\mathcal{D}_t$ (a set of groups at time step t). After knowing the group structure, we get x_{ij} for all i, j, and then we can solve the P_b and P_g maximizing (2). In this way, we can verify how good the overlapping algorithm works with the communication model.

3.2 Learning from Group Structure

From the previous stage, we have a set of group structures $\mathcal{D}_t$, $t = 1 \cdots T$. However, in order to use the learning method in next stage, one needs to construct the paths of each actor. This means we need the correspondence between groups of time step t and $t+1$, in order to determine actors' actions. Formally, we need a matching between the groups in $\mathcal{D}_t$ and $\mathcal{D}_{t+1}$ for $t = 1, \ldots, T-1$: for each group in $\mathcal{D}_t$, we must identify the corresponding group in $\mathcal{D}_{t+1}$ to which it evolved. If there are more groups in $\mathcal{D}_{t+1}$, then some new groups arose. If there are fewer groups in D_{t+1}, then some of the existing groups disappeared. In order to find this matching, we use a standard greedy algorithm.

Finding Matchings. Let $\mathcal{X} = \{X_1, \ldots, X_n\}$ and $\mathcal{Y} = \{Y_1, \ldots, Y_n\}$ be two collections of sets, and we allow some of the sets in $\mathcal{X}$ or $\mathcal{Y}$ to be empty. We use the symmetric set difference $d(x,y) = 1 - |x \cap y|/|x \cup y|$ as a measure of error between two sets. Then, we consider the complete bipartite graph on $(\mathcal{X}, \mathcal{Y})$ and would like to construct a matching of minimum total weight, where the weight on the edge (X_i, Y_j) is $d(X_i, Y_j)$. This problem can be solved in cubic time using max-flow techniques [10]. However, for our purposes, this is too slow, so we use a simple greedy heuristic. First find the best match, i.e. the pair (i^*, j^*) which minimizes $d(X_i, Y_j)$ over all pairs (i, j). This pair is removed from the sets and the process continues. An efficient implementation of this greedy approach can be done in $O(n^2 \log n)$, after $d(X_i, Y_j)$ has been computed for each pair (i, j).

3.3 Learning from Group Evolution

We first introduce some notation. The set of actors is $\mathcal{A}$; we use i, j, k to refer to actors. The data $\mathcal{D} = \{\mathcal{D}_t\}_{t=1}^{T+1}$ is the set of social groups at each time step, where each $\mathcal{D}_t$ is a *set* of groups, $\mathcal{D}_t = \{G_l^t\}_l$, $G_l^t \subseteq \mathcal{A}$; we use l, m, n to refer to groups. Collect all the parameters which specify the model as $\boldsymbol{\Theta}_M$, which includes all the parameters specific to an actor (*e.g.,* $type$) and all the global parameters in the model (*e.g.,* $\theta_{action}, \theta_{reward}, \theta_{group}$). We would like to maximize the likelihood, $\mathcal{L}(\boldsymbol{\Theta}_M) = Prob(\mathcal{D}|\boldsymbol{\Theta}_M)$. We define the path of actor i, $\mathbf{p}_i^T = (p_i(1), \ldots, p_i(T))$, as the set of actions it took over the time steps $t = 1, \ldots, T$. The actions at time t, $p_i(t)$, constitute deciding to join, leave or stay in groups, as well as which groups were left or joined. Given $\mathcal{D}$, we can construct $\mathbf{p}_i^T$ for every actor i, and conversely, given $\{\mathbf{p}_i^T\}_{i=1}^{|\mathcal{A}|}$, we can reconstruct $\mathcal{D}$. Therefore, we can alternatively maximize

$$\mathcal{L}(\boldsymbol{\Theta}_M) = Prob(\mathbf{p}_1^T, \ldots, \mathbf{p}_{|\mathcal{A}|}^T \| \boldsymbol{\Theta}_M). \tag{3}$$

It is this form of the likelihood that we manipulate. Typical ways to break up this optimization is to iteratively first improve the continuous parameters and then the combinatorial (discrete) parameters. The continuous parameters can be optimized using a gradient based approach, which involves taking derivatives of $\mathcal{L}(\boldsymbol{\Theta}_M)$. This is generally straightforward, though tedious, and we do not dwell on the technical details. The main problem we face is an algorithmic one, namely that typically, the number of actors, $|\mathcal{A}|$ is very large (thousands or tens of thousands), as is the number of time steps, T, (hundreds). From the viewpoint of actor i, we break down $\boldsymbol{\Theta}_M$ into three types of parameters: θ_i, the parameters specific to actor i, in particular its type and initial capital; $\theta_{\bar{i}}$, the parameters specific to other actors; and, θ_G, the parameters of the society, global to all the actors. The optimization style is iterative in the following sense. Fixing parameters specific to actors, one can optimize with respect to θ_G. Since this is a fixed number of parameters, this process is algorithmically feasible. We now consider optimization with respect to θ_i, fixing $\theta_{\bar{i}}, \theta_G$. This is the task which is algorithmically non-trivial, since there are $\Omega(|\mathcal{A}|)$ such parameters.

In our model, the actors at each time step take independent actions. At time t, the state of the society $\mathcal{I}_t$ can be summarized by the group structure, the actor ranks in each group and the actor surplus resources. Given $\mathcal{I}_t$, each actor acts independently at time t. Thus, we can write

$$\mathcal{L}(\boldsymbol{\Theta}_M) = Prob(\mathbf{p}_1^{T-1}, \ldots, \mathbf{p}_{|\mathcal{A}|}^{T-1} | \boldsymbol{\Theta}_M) \times \prod_{i=1}^{|\mathcal{A}|} Prob(p_i(T) | \boldsymbol{\Theta}_M, \mathcal{I}_T). \tag{4}$$

Continuing in this way, by induction,

$$\mathcal{L}(\boldsymbol{\Theta}_M) = \prod_i \prod_t Prob(p_i(t) | \boldsymbol{\Theta}_M, \mathcal{I}_t). \tag{5}$$

The actions of $\bar{i} \neq i$ depends on θ_i only through $\mathcal{I}_t$, which is a second order dependence, therefore we ignore the second term in optimizing the parameters specific to actor i, and take logarithm,

$$\theta_i^* \leftarrow \text{argmax} \sum_t \log Prob(p_i(t) | \boldsymbol{\Theta}_M, \mathcal{I}_t). \tag{6}$$

Thus the maximization over a single actor's parameters only involves that actors path and is a factor of $|\mathcal{A}|$ more efficient to compute than if we looked at all the actor paths.

Therefore, the entire learning process can be summarized by maximizing over each parameter successively, where to maximize over the parameters specific to an actor, we use only that actor's path.

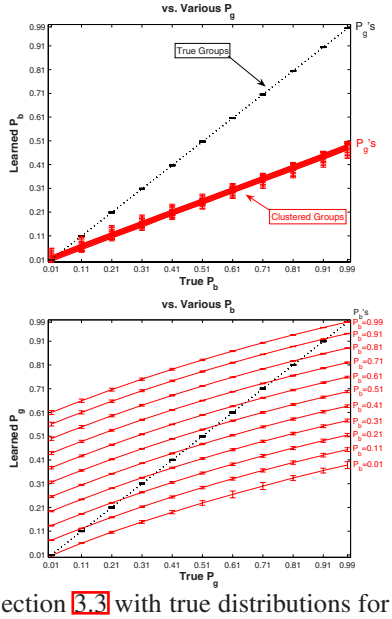
4 Experiments

In our model, there are a lot of parameters which can be learned, however, here we show the results about learning communication probabilities and actors' type from the synthetic data and also from real data, such as Enron email and movie newsgroups.

4.1 Results on Synthetic Data

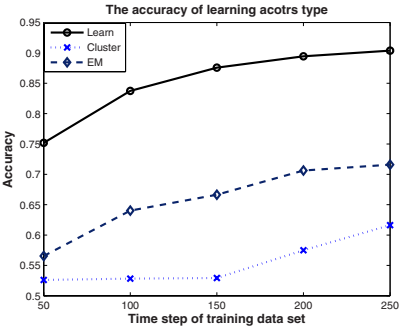
To evaluate performance, we use an instance of the model to generate synthetic data for training and testing. Since we know the values of parameters in the model, we can compare the true type with the learned type to compute the accuracy. We simulate 50, 100, 150, 200 and 250 time steps of training data (averaged over 20 data sets). Each data set was created by randomly assigning about 1/3 of the actors to each class. All others parameters except types and distributions of group size preference were held fixed.

Learning Communication Probabilities. The figures at right show the results of learning communication probabilities. When the group structures are known, the algorithm can learn the P_b and P_g very well (dot lines). Meanwhile, as we apply the overlapping clustering algorithms in [4] to get the group structures, the upper figure show the learned P_b's are not influenced by different P_g's. However, different P_b's have an impact on the learned P_g's (shown at the lower figure) because some outsiders have been included in the group. The bottom line of learning communication probabilities is finding the monotonic relationship.



Learning Actors' Type. We evaluate the learning results from the following 3 different algorithms (details can be found in [6]):

- Learn: The learning algorithm described in Section 3.3 with true distributions for group size preference.
- Cluster: For each actor i, let $size_i$ be the average size of groups actor i joined. We cluster $\{size_i\}_{i=1}^{|\mathcal{A}|}$ into 3 groups using the standard 3-means algorithm. This is a simple heuristic based on the observation that leaders join small groups and followers large groups.
- EM: With unknown distributions for group size preference, we use expectation-maximization (EM) algorithm cooperating with Learn and Cluster to learn the actors' type as well as the distributions for group size preference.



The figure on the right hand side shows the accuracy (%) of Learn, Cluster and EM algorithms with different time steps of training data set, and for comparison, the accuracy of randomly assigning type is 0.33. The results tell that the accuracy for Learn algorithm is the best because it uses the true distribution of group size preference and only need to learn the actors' type. The Cluster algorithm has the worst result because it only considers the group size preference and omits the interactions with other actors. The EM algorithm learn actors' type also the distribution of group size preference. The figure tell that the EM algorithm does improve the results from the Cluster algorithm. The Cluster algorithm is only based on the average size of groups the actor joined which can be detected from *observable* group evolution data. On the other hand, the Learn and EM algorithms learn actors' *hidden* curriculum based on the interactions with other actors and the influences of the environment, which cannot be observed from the data. From the results, we also can tell when the length of the time period of training data set increases, we obtain better results from all algorithms. The reason is that more data points we can learn from, more accuracy of the results we can achieve.

4.2 Results on Real Data

Our results on real data are based on communications because it is difficult to collect data that includes the group evolution from the real world. Hence, we use the multistage learning process and algorithms in Section 3 to learn the parameters.

Movie Newsgroup. We collected the communication data from a movie newsgroup, which includes 1528 active actors. We apply both EM and Cluster algorithms on the data set, and the results are shown in the table below. Based on Cluster algorithms, the

	Learned Actors' Types		
	Leader	*Socialite*	*Follower*
Number of Actor	822	550	156
Percentage	53.8%	36.0%	10.2%

(a) Cluster algorithm

	Learned Actors' Types		
	Leader	*Socialite*	*Follower*
Number of Actor	532	368	628
Percentage	34.8%	24.1%	41.1%

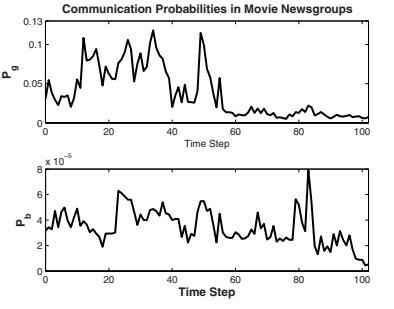
(b) EM algorithm

majority of actors are leaders which *only* meant that they joined the small groups – yet, this does not represent these actors' preferences in group size. This result of Cluster algorithm in which group size that actors joined proves to match the finding that was done by hands in Butler's social analysis in the newsgroups data [11]. However, the result of EM algorithm shows that the number of *Follower* increases 30.9%, the number of *Leader* decreases 19%, and the number of *Socialite* decreases 11.9%.

According to the research data shown as above, there is a significant difference between both results: in Butler's finding (Cluster algorithms), it is easily for one to locate which size of the groups an actor joined manually but it is difficult for one to detect the

actor's actual group size preferences, *i.e.*, social interactions between actors can play an influential role in the actors' decision making. By applying EM algorithm approach, one can not only consider the observable groups size that an actor joined but also the social interactions between the actors. As can be seen from above data, the approach in using EM algorithm yields similar result that shows the majority of actors would more likely to read news (*Follower*) than to post news (*Leader*) in a movie newsgroup community.

The figures at the right show the learned the communication probabilities at different time step, P_g and P_b. A comparison from both figures show that people in the same group communicate more frequently than people in the different group (P_b is much smaller than P_g). In addition, from the upper figure, we see two ranges of obvious activities - more active communications and much reduced communications.



Enron Email. By using the strategies in [12] to cleaning Enron email data set from November 13th, 1998 to June 21st, 2002 and obtain the communication network for 154 active actors, we are able to obtain learning results of EM and Cluster. Both of the EM and Cluster results are very similar (table shown below). The reason being

	Learned Actors' Types		
	Leader	*Socialite*	*Follower*
Number of Actor	28	50	76
Percentage	18.2%	32.5%	49.3%

(a) Cluster algorithm

	Learned Actors' Types		
	Leader	*Socialite*	*Follower*
Number of Actor	24	62	68
Percentage	15.6%	40.2%	44.2%

(b) EM algorithm

that in a company, an individual's preference is usually masked because the employees cannot change their "jobs" as freely as their responsibilities will change accordingly. Yet, in the movie newsgroups, actors can change groups anytime according to one's desire. Therefore, the communications within Enron email network are based upon the need of work, and employees (*Socialite* or *Follower*) cannot just join a group due to the attraction of the manager (*Leader*) of that group.

5 Conclusions

We have presented a parameterized agent-based hidden Markov model for learning actors' dynamics and the micro-laws governing the society's social group dynamics. The benefits of the multistage learning process are to extracting different information about the actor and the society dynamics, to reduce the learning noise, and to setup the checking point for evaluating the performance of algorithms, at each learning stage. Our main contributions are the application of efficient algorithms and heuristics toward learning the parameters in the specific application of modeling social groups and communications. Our results on synthetic data indicate that when the model is well specified, the learning is accurate. Since the model is sufficiently general and grounded in social science theory, any given instance of the model can be appropriate for a given society.

Therefore, under this stance, almost any general model of this form which is founded in social science theory will yield outputs that can serve as productive reference to one's decision making or stimulating triggers to new research studies.

Acknowledgments. This material is based upon work partially supported by the U.S. National Science Foundation (NSF) under Grant Nos. IIS-0621303, IIS-0522672, IIS-0324947, CNS-0323324, NSF IIS-0634875 and by the U.S. Office of Naval Research (ONR) Contract N00014-06-1-0466 and by the U.S. Department of Homeland Security (DHS) through the Center for Dynamic Data Analysis for Homeland Security administered through ONR grant number N00014-07-1-0150 to Rutgers University. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the position or policy of the National Science Foundation, Office of Naval Research, Department of Homeland Security, or the U.S. Government, and no official endorsement should be inferred or implied.

References

1. Wasserman, S., Faust, K.: Social Network Analysis. Cambridge University Press, Cambridge (1994)
2. Goldberg, M., Horn, P., Magdon-Ismail, M., Riposo, J., Siebecker, D., Wallace, W., Yener, B.: Statistical modeling of social groups on communication networks. In: First conference of the North American Association for Computational Social and Organizational Science (2003)
3. Backstrom, L., Huttenlocher, D., Kleinberg, J., Lan, X.: Group formation in large social networks: Membership, growth, and evolution. In: KDD 2006: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 44–54 (2006)
4. Baumes, J., Goldberg, M., Magdon-Ismail, M.: Efficient identification of overlapping communities. In: IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 27–36 (2005)
5. Berger-Wolf, T.Y., Saia, J.: A framework for analysis of dynamic social networks. In: KDD 2006: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 523–528. ACM Press, New York (2006)
6. Chen, H.C., Goldberg, M., Magdon-Ismail, M., Wallace, W.A.: Inferring agent dynamics from social communication network. In: Joint 9th WebKDD and 1st SNA-KDD Workshop at KDD (2007)
7. Baum, L.E., Sell, G.R.: Growth functions for transformations on manifolds. Pacific Journal of Mathematics 27(2), 211–227 (1968)
8. Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. IEEE Transactions on Information Theory IT-13(2), 260–269 (1967)
9. Baum, L.E., Egon, J.A.: An inequality with applications to statistical estimation for probabilistic functions of a markov process and to a model for ecology. Bulletin of the American Mathematical Soc. 73, 360–363 (1967)
10. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. Mcgraw-Hill, Cambridge, MA (2001)
11. Butler, B.: The dynamics of cyberspace: Examining and modeling online social structure. Technical report, Carnegie Melon University, Pittsburgh, PA (1999)
12. Zhou, Y., Goldberg, M., Magdon-Ismail, M., Wallace, A.: Strategies for cleaning organizational emails with an application to enron email dataset. In: 5th Conf. of North American Association for Computational Social and Organizational Science (2007)

Effects of Neighbourhood Structure on Evolution of Cooperation in N-Player Iterated Prisoner's Dilemma

Raymond Chiong[1], Sandeep Dhakal[1], and Lubo Jankovic[2]

[1] School of IT & Multimedia, Swinburne University of Technology (Sarawak Campus), Kompleks Negeri, Jalan Simpang Tiga, 93576 Kuching, Sarawak, Malaysia
rchiong@swinburne.edu.my, sandeep.dhakal@gmail.com
[2] InteSys Ltd, Birmingham Research Park, Vincent Drive, Birmingham B15 2SQ, UK
L.Jankovic@e-intesys.com

Abstract. In multi-agent systems, complex and dynamic interactions often emerge among individual agents. The ability of each agent to learn adaptively is therefore important for them to survive in such changing environment. In this paper, we consider the effects of neighbourhood structure on the evolution of cooperative behaviour in the N-Player Iterated Prisoner's Dilemma (NIPD). We simulate the NIPD as a bidding game on a two dimensional grid-world, where each agent has to bid against its neighbours based on a chosen game strategy. We conduct experiments with three different types of neighbourhood structures, namely the triangular neighbourhood structure, the rectangular neighbourhood structure and the random pairing structure. Our results show that cooperation does emerge under the triangular neighbourhood structure, but defection prevails under the rectangular neighbourhood structure as well as the random pairing structure.

Keywords: N-player iterated prisoner's dilemma, bidding game, game strategies, neighbourhood structure.

1 Introduction

The Iterated Prisoner's Dilemma (IPD) game has been used extensively to study the evolution of cooperative behaviour over the last two decades [1-4]. However, the game is somewhat limited as it is only between two players. Many real world problems, especially the social and economic ones, often require interactions among various parties. As such, an extension to the IPD game which is more realistic can be found in the multiplayer version game called N-Player IPD (NIPD) [5-6].

NIPD is a collective game that involves more than two players. Each player can either cooperate or defect against other players, getting a reward which increases when there are more players cooperating. This allows the simulations of many aspects of the real world societies, where one can choose to help the population as a whole or help himself at the expense of others.

In this paper, we consider the effects of neighbourhood structure on the evolution of cooperative behaviour in the NIPD game. Previous works have studied the impact

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 950–959, 2007.

of the number of players [6], payoff function [7], neighbourhood size [7], history length [8], localisation issue [8], population structure [9], generalisation ability [6, 10], forgiveness [11], trust [12], cultural learning [13], noise [14], etc but none has investigated the influence of the neighbourhood structure on the evolution of cooperation among players in the context of NIPD. We simulate the NIPD as a multi-agent bidding game using a two dimensional grid-world, where each agent is required to bid either high or low against its neighbours based on a chosen game strategy. Every agent on the grid-world competes with its neighbours iteratively and gets its scores from their interactions. During the process, the agent looks at its neighbours' scores. If no neighbour has a higher score, the agent retains its original strategy. If the agent has a neighbour or neighbours with higher scores, it will learn to adopt the strategy of the most successful neighbour. With such arrangement, an agent may benefit from actions of other agents whom it interacts with, and it may also take actions that benefit the other agents. The willingness of an agent to take actions which are beneficial to another agent depends strongly on its previous interactions with that agent.

We introduce three different types of neighbourhood structures in which the agents play the bidding game, namely the triangular neighbourhood structure, the rectangular neighbourhood structure and the random pairing structure. We examine these neighbourhood structures with four agents (i.e. N = 4) playing in a group each time. We expect different neighbourhood structures to affect the outcome of the game differently.

The rest of this paper is organised as follows: Section 2 introduces the background of NIPD. In Section 3, we present the methodology we use for this work. Section 4 describes the experimental setting and results. Finally, we draw conclusion in Section 5 and highlight potential future work.

2 Background

The NIPD is an extension of the conventional IPD game in which a group of N players, where N is greater than 2, will continuously interact with each other rather than only two playing against one another. According to Davis et al. [15], NIPD has greater generality and applicability to real life situations than IPD, as many of the real life problems can be represented with the NIPD paradigm in a much realistic way.

In an NIPD game, N players make decision independently on two actions, either cooperate or defect, without knowing other players' choices. Let cooperate be C and defect be D, each player attains the score based on the payoff functions $C(c) = c/N$ for cooperation and $D(c) = (c + k)/N$ for defection, where c is the number of cooperators and k is the total cost for cooperation ($1 < k < N$) [9]. Players are normally engaged with each other iteratively, competing for higher average scores.

NIPD is also different from the IPD in the kind of strategies that players can adopt successfully. Various strategies that thrive in punishing the defectors in an IPD game, for example Tit For Tat (TFT), a very simple strategy and overall winner of both Axelrod's IPD tournaments [1-2], would not necessarily be so successful in an NIPD game. Punishment handed out to defectors would result in unintentional punishment of other cooperators in the group as well, thus rendering cooperation meaningless to a

certain extent. This makes the NIPD game more complex and interesting than the IPD game.

3 Methodology

The main goal of our work is to examine the effects of neighbourhood structure on the evolution of cooperative behaviour in the NIPD game. In doing so, we construct a two dimensional grid-world in the form of cellular automata to simulate a multi-agent bidding game. Each agent has two actions of either to bid high or to bid low. Every agent will play the bidding game against its neighbours in a group of four (i.e. N = 4), and their actions will be determined by the strategy they choose. The outcome of the game is decided by the payoff shown in Table 1 below.

Table 1. The payoff for our bidding game with N = 4

Bid Results	Payoff
4 high	-1
3 high 1 low	1 -3
2 high 2 low	3 -1
1 high 3 low	5 1
4 low	3

From Table 1 we see that bidding high is dominant for each agent, because bidding high always yields a better payoff than bidding low no matter how many of the other agents bid low. However, should all agents choose the non-dominant low bids, the outcome would be much better for the group as it yields a total payoff of 12, the highest total payoff among the overall bid results.

3.1 The Cellular Automata Model

Our world is implemented in the form of cellular automata on a 20 x 20 grid, within a C++ program to simulate agents with different strategies. The grid is formed by a two dimensional array with overlapping edges, which means every cell on the grid has eight immediate neighbouring cells, including those at the edges of the grid. Each agent occupies one cell on the grid, thus there are 400 agents in total in the grid-world. Different colours are used for the cells to represent different strategies adopted by the agents. Each agent is designed to compete against its three immediate neighbours with its initial strategy. Scores are calculated continuously while the game is in progress, and the agents with less successful strategies will start learning the more successful strategies adopted by the neighbouring agents.

We introduced three different neighbourhood structures in our experiments, namely the triangular neighbourhood structure, the rectangular neighbourhood structure and the random pairing structure. Figure 1 shows the spatial interactions among agents with the three different types of neighbourhood structures. It is

necessary to note that for random pairing structure, the interactions among agents in a group are not confined to the eight immediate neighbours only, but any two agents throughout the grid-world can be paired at random.

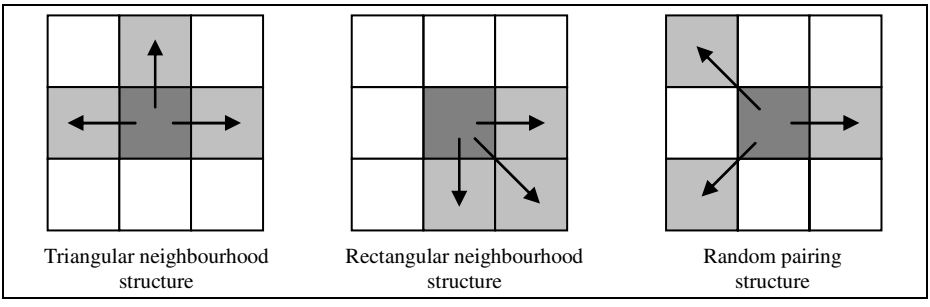


Fig. 1. Three different types of neighbourhood structures

3.2 Strategy Representation

The game strategies that determine the agents' moves can be represented in many different ways. A good representation is always important for this kind of study to ensure the accuracy of results. After much consideration, we have decided to adopt the representation developed by Yao and Darwen [6] as it is exponentially much shorter and easier to implement.

Under this representation, a history of l rounds for an agent can be represented as the combination of the following:

- l bits to represent the agent's l previous bids. Here, a '1' indicates a high bid, and a '0' indicates a low bid.
- $l * \log_2 N$ bits to represent the number of low bidders in the previous l rounds among the agent's N-1 group members. Here, N is the group size.

In this paper, our group size is 4. We have limited the number of previous bids in memory to 3 (i.e. $l = 3$). Therefore, based on the above representation scheme, the history for an agent would be $3 + 3 \log_2 4 = 9$ bits long.

For example, an agent in our bidding game could have a history as follows:

110 11 01 10

Here, the first 3 bits are the agent's previous three moves. This means that the agent bid high in the last 2 rounds, and bid low three rounds before. The subsequent 6 bits indicate that all its group members, i.e. N-1 = 3, bid low in the last round, one group member bid low in the round before last round, and two group members bid low three rounds ago.

Since each history bit string is 9 bits long, there are $2^9 = 512$ possible histories in our game. Due to the fact that there is no memory of previous rounds at the beginning of the game, the agents' history would not be 9 bits long for the first three rounds. They would normally be 3 and 6 bits long at the beginning of the 2^{nd} and 3^{rd} rounds respectively.

As each strategy we employ in the game needs to have a response for every possible history, each strategy is at least 512 bits long. Depending on different strategies, some strategies would require a maximum of three predefined bids when the history is less than the required 9 bits in length. Hence, our strategies generally range from 512 to 515 bits in length.

3.3 Strategy Choice

After devising a way for the strategy representation, an initial population of agents with different strategies needs to be determined. It is impossible for us to include all the game strategies from the entire history of the IPD literature in our simulation experiments, therefore only twelve strategies are being chosen, some of which are distinctive, and some are intelligent. We present these twelve strategies in the N-player versions. For example, an N-player version of TFT will bid low on the first move and then bid whatever the majority of its group members bid in the previous round. Table 2 summarises all the twelve strategies in N-player versions.

We distribute the twelve strategies throughout the grid-world, with each agent acquiring its own initial strategy to play the bidding game. Based on the selection of strategies, every agent starts bidding either high or low during each interaction. Various colours are used to represent different strategies selected by different agents. When an agent learns and adopts another strategy from a more successful agent in order to win the game, the colour of the cell where the agent resides will change, reflecting the new strategy of its choice.

Table 2. The twelve strategies in N-player versions

Strategies	Description
All_High	Bids high all the time.
All_Low	Bids low all the time.
High_Low	Takes turns to bid high and low. Bids high the first time.
Low_High	Takes turns to bid high and low. Bids low the first time.
Per_hhl	Bids in the sequence [high, high, low].
Per_llh	Bids in the sequence [low, low, high].
Tit For Tat (TFT)	Bids low on the first move and then bids what the majority of its neighbours bid in the previous round.
Prober	Begins by bidding [low, high, high]. Continues to bid high if the majority of the opponents bid low in the previous two rounds. Otherwise, plays TFT.
Mistrust	Bids high on the first move and then bids what the majority of its neighbours bid in the previous round.
Pavlov	Bids high on the first move and then bids low only if the majority of its neighbours bid the same as itself.
TF2T	Bids low from the beginning. Bids high only if the majority of its neighbours bid high for two consecutive rounds.
Spiteful	Bids low until the majority of its neighbours bid high, then bids high all the time.

4 Experiments and Results

In this section, we present experiments undertaken and discuss the corresponding results based on the twelve strategies described in the previous section. As our objective is to examine the effects of the neighbourhood structure on the evolution of cooperative behaviour in the bidding game, three separate sets of experiments are conducted on the different types of neighbourhood structures. For all our experiments, we use the population size of 400 agents in which all the agents are inhibited on a 20 x 20 grid-world. The twelve strategies selected are randomly distributed among the agents, as evenly as possible, in the beginning of the experiments (see Figure 2). All the experiments are designed for all agents to play against one another iteratively with a group size of four for 200 generations. In every generation, each agent evaluates its neighbours' average scores. If no neighbour has a higher average score, the agent retains its original strategy. If the agent has a neighbour or neighbours with higher average scores, it will learn to adopt the strategy of the most successful neighbour. Scoring in the game is based on the payoff mentioned in Table 1.

It is necessary to highlight here that the average scores are used to evaluate the agents' performance rather than individual scores. This is because individual score only tells us how well an agent is doing as an individual, but average score tells us how well it is doing in its group. Individual scores may be used in an IPD game but are not so suitable for an NIPD game.

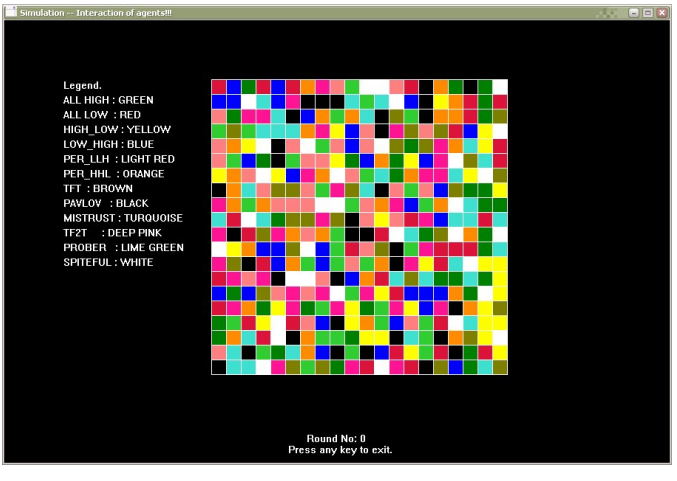


Fig. 2. Initial distribution of the twelve strategies in the beginning of the experiment

4.1 Experimental Results on Triangular Neighbourhood Structure

We have shown earlier in Figure 1 that in the triangular neighbourhood structure, each agent has three group members with whom it interacts, one immediately above itself, one to its left and another to its right. Meanwhile, the agent's group members will also have other different groups associating to them. This allows overlapping between different groups of agents during the game.

In our first experiment with the triangular neighbourhood structure, we saw that All_Low and strategies that reciprocate against the opponents were dominant. Non-cooperative strategies like All_High, Per_hhl, etc disappeared early in the game. This means that the agents are preferring cooperation over defection under this neighbourhood structure. From Figure 3 we observed that even though there were equal or near equal amount of cooperators and defectors in the beginning, almost all the agents were cooperating after 20-30 generations. In most cases, there would be no defectors left at that point.

We believe that the emergence of cooperation via the triangular neighbourhood structure is due to the ability of the agents to reciprocate cooperation and retaliate against defection. Owing to the overlapping nature of the structure, every agent who is interacting with its own group members is also a member of the groups of its neighbours to the left and the right. This means that the agent can indirectly influence the average group scores of two of its neighbours, thus significantly increases the agent's bargaining power against its group members, thereby leading to cooperation.

Fig. 3. The number of cooperators (green) *vs.* defectors (red) in triangular neighbourhood structure

4.2 Experimental Results on Rectangular Neighbourhood Structure

In the case of the rectangular neighbourhood structure, each agent has one group member to its right, one below itself and another one diagonally below it to the right. As with the triangular structure, the group of the agent and the groups of its neighbours are overlapping. However, unlike the triangular structure, the agent itself does not belong to any of the groups of its neighbours.

In our second experiment with the rectangular neighbourhood structure, we noticed that Per_hhl and strategies that tend to follow their opponents did significantly well. Cooperative strategies mostly disappeared early in the game, and even All_High was not doing so well. Although most of the agents were bidding high regularly, they occasionally bid low as well. Only at a few instances was All_High more dominant than other strategies. Even in those instances there was always a strong presence of strategies which would bid low occasionally like Per_hhl, high_low, etc. The graph in

Figure 4 evidenced this fact that even though most of the agents were bidding high customarily, there were always some agents who were bidding low. In a few cases, the population of cooperators was as high as one fourth of the entire population.

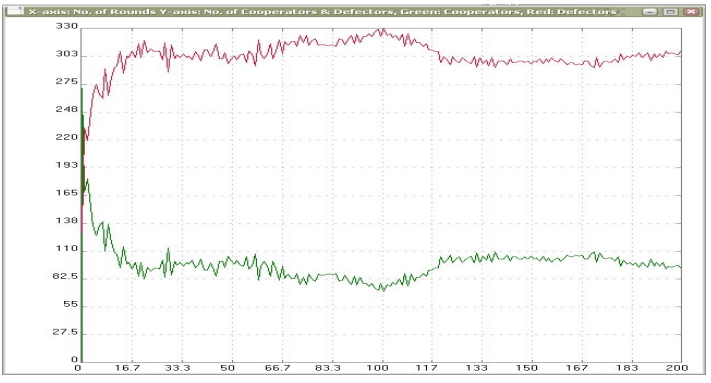


Fig. 4. The number of cooperators (green) *vs.* defectors (red) in rectangular neighbourhood structure

The result with rectangular neighbourhood structure is rather sluggish as compared to the triangular neighbourhood structure. Nevertheless, defective behaviour is favoured as the agents do not have much chance of retaliating against defectors. This minimises the impact of the significance given to the group rationality, allowing agents to promote their own interests. Still, the emphasis given to group interests in the game ensures that a small percentage of cooperators always existed. The cooperators, though their individual scoring is being hurt by the defectors, would not look to defect because their groups are doing as well as others in most cases. They cannot induce cooperation, though, because they lack the power to retaliate. The defectors, on the other hand, keep on defecting, without any fear of retaliation.

4.3 Experimental Results on Random Pairing Structure

In random pairing structure, each agent's group members are selected randomly from the population. The groups would change dynamically after each generation is played. In comparison with the triangular and rectangular structures, where each agent is a member of four overlapping groups, the random pairing allows an agent to be selected in a maximum of four groups. Due to the random pairing nature, the chances that there is overlap between the agent's group and the groups of its group members are minimal.

In our third experiment with the random pairing structure, we observed that defection was rampant among the population from very early in the game. The grid would normally show no changes in the agents' strategies after about 20-30 generations. All the cooperative strategies vanished before they could reach the 30th generation. All_High was quite dominant along with the likes of TFT, Mistrust, Spiteful, etc. All the latter strategies were second best to All_High in no particular order. The strength of their performance was also arbitrary. Figure 5 shows the result where, by around 30 generations onwards, there were literally no more cooperators

left in the game. Since all the agents were defecting and getting similar scores, there was no need for a strategy change.

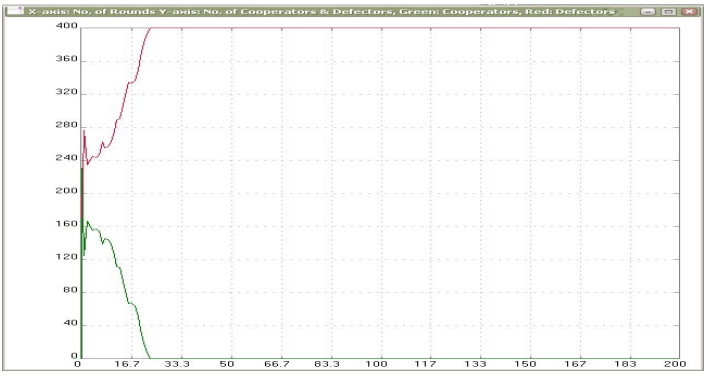


Fig. 5. The number of cooperators (green) *vs.* defectors (red) in random pairing structure

It is easy to see why defection is so dominant in the case of random pairing. Due to the randomly pitted interactions, anonymity is often guaranteed as the chances of the same group members meeting one another again are extremely small. As there is no fear of reciprocation or retaliation, there is little pressure for cooperative behaviour. This result concurs with Ishibuchi and Namikawa [16] who used a similar random pairing scheme in their studies on evolving IPD game strategies under structured demes. They demonstrated that the evolution of reciprocal strategies is very difficult to achieve when opponents are selected randomly.

5 Conclusion

In this paper, we investigated the effects of three different neighbourhood structures on the emergence of cooperation in a multi-agent bidding game. Our experimental results demonstrated that agents need to have the ability to punish defection in order to induce cooperation from a population with numerous defectors. This occurs in overlapping neighbourhoods in which one agent is at a center of one neighbourhood, and at a fringe of another. In this way the behaviour of each agent influences more than one neighbourhood, and results in higher memory of the system and higher cooperation. That is why cooperation emerges in the triangular neighbourhood structure but not in the rectangular and the random pairing neighbourhood structures. We also showed that without the fear of being retaliated against, defectors and players with opportunistic strategies would make use of the cooperators around them only to forward their self-interests.

In our current work the agents are limited to choosing only from a fixed set of strategies and they lack the ability to alter their strategies. However, alteration rather than complete abandonment of their strategies would be closer to the rules of nature and reality. Therefore, the future work will involve the use of co-evolutionary learning within the population. This would enable us to see whether or not co-

evolution can induce the emergence of cooperation in less favourable neighbourhood structures such as the rectangular or random pairing.

References

1. Axelrod, R.: The Evolution of Cooperation. Basic Books, New York (1984)
2. Axelrod, R.: The Evolution of Strategies in the Iterated Prisoner's Dilemma. In: Davis, L. (ed.) Genetic Algorithms and Simulated Annealing, pp. 32–41. Morgan Kaufmann, Los Altos, CA (1987)
3. Lindgren, K.: Evolution Phenomena in Simple Dynamics. In: Langton, C.G., Taylor, C., Farmer, J.D., Rasmussen, S. (eds.) Artificial Life II, pp. 295–312. Addison-Wesley, Reading, MA (1991)
4. Fogel, D.B.: Evolving Behaviors in the Iterated Prisoner's Dilemma. Evolutionary Computation 1(1), 77–97 (1993)
5. Bankes, S.: Exploring the Foundations of Artificial Societies: Experiments in Evolving Solutions to Iterated N-player Prisoner's Dilemma. In: Brooks, R.A., Maes, P. (eds.) Artificial Life IV, pp. 237–242. MIT Press, Cambridge, MA (1994)
6. Yao, X., Darwen, P.: An Experimental Study of N-person Iterated Prisoner's Dilemma Games. Informatica 18(4), 435–450 (1994)
7. Seo, Y.G., Cho, S.B., Yao, X.: The Impact of Payoff Function and Local Interaction on the N-player Iterated Prisoner's Dilemma. Knowledge and Information Systems: An International Journal 2(4), 178–461 (2000)
8. Seo, Y.G., Cho, S.B., Yao, X.: Emergence of Cooperative Coalition in NIPD Game with Localization of Interaction and Learning. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 877–884. IEEE Press, Piscataway, NJ (1999)
9. Suzuki, R., Arita, T.: Evolutionary Analysis on Spatial Locality in N-Person Iterated Prisoner's Dilemma. International Journal of Computational Intelligence and Applications 3(2), 177–188 (2003)
10. Yao, X.: Automatic Acquisition of Strategies by Co-evolutionary Learning. In: Proceedings of the International Conference on Computational Intelligence and Multimedia Applications, Gold Coast, Australia, pp. 23–29. Griffith University Press, Brisbane, Australia (1997)
11. O'Riordan, C., Griffith, J., Newell, J., Sorensen, H.: Co-evolution of Strategies for an N-player Dilemma. In: Proceedings of the 2004 Congress on Evolutionary Computation, pp. 1625–1630. IEEE Press, Piscataway, NJ (2004)
12. Birk, A.: Trust in an N-player Iterated Prisoner's Dilemma. In: Proceedings of the International Conference on Autonomous Agents, 2 Workshop on Deception, Fraud and Trust in Agent Societies, Seattle, WA (1999)
13. O'Riordan, C., Griffith, J., Curran, D., Sorensen, H.: Norms and Cultural Learning in the N-player Prisoner's Dilemma. In: Proceedings of the 2006 Congress on Evolutionary Computation, pp. 1105–1110. IEEE Press, Piscataway, NJ (2006)
14. Chong, S.Y., Yao, X.: Behavioral Diversity, Choices and Noise in the Iterated Prisoner's Dilemma. IEEE Transactions on Evolutionary Computation 9(6), 540–551 (2005)
15. Davis, J.H., Laughlin, P.R., Komorita, S.S.: The Social Psychology of Small Groups. Annual Review of Psychology 27, 501–542 (1976)
16. Ishibuchi, H., Namikawa, N.: Evolution of Iterated Prisoner's Dilemma Game Strategies in Structured Demes under Random Pairing in Game Playing. IEEE Transactions on Evolutionary Computation 9(6), 552–561 (2005)

Interface Agents' Design for a DRT Transportation System Using PASSI

Claudio Cubillos and Sandra Gaete

Pontificia Universidad Católica de Valparaíso
Escuela de Ingeniería Informática
Av. Brasil 2241, Valparaíso, Chile
claudio.cubillos@ucv.cl, sandra.gaete@gmail.com

Abstract. The present work continues a longer research in the field of flexible transportation services and the design of an agent system devoted to the planning, scheduling and control of trips under such a domain. In particular, this paper focuses in the design and development of the interface agents present in the system by following an agent development methodology named PASSI. The interface agent devoted to interaction with the customers is explained in detail and its prototype is shown.

1 Introduction

In the last two decades, the mobility needs of European citizens have radically changed. The need to cover more diffuse travel patterns, varying periods of low demand, city-peripheral journeys, as well as commuting trips has leveraged the need of Demand-Responsive Transport services (DRTS) in which routes, departure times, vehicles and even operators, can be matched to the identified demand allows a more user-oriented and cost effective approach to service provision.

Software agents are defined as autonomous entities capable of flexible behavior denoted by reactiveness, pro-activeness and social ability [1]. Multiagent systems (MAS) consist of diverse agents that communicate and coordinate generating synergy to pursue a common goal. In this context the present work describes the design of a multiagent system using the agent development methodology called PASSI [2] for modeling a passenger transportation under a flexible approach. In this way, it gives continuity to our past research [3] [4] on heuristics for solving scheduling of passenger trips. In particular the paper focuses in describing the design of the interface agents for the main actors; Customer and Driver, for then detailing the interface agent prototype devoted to Customers.

The paper scope moves toward the design description of interface agents using PASSI withing the context of a complete system. Although literature is plenty of agent systems, agent software engineering (AOSE) is less common. In the particular case of PASSI, finding complete designs in addition to the examples developed by the own creators is not an easy task. Furthermore, a practical design with PASSI devoted to interface agents, showing how interface events are tackled and mapped in terns of tasks, roles, the granularity required, etc. is not

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 960–969, 2007.

present in literature (at least at the best of our knowledge). Therefore our aim is to somehow cover this lack.

2 Related Work

The research on Multi-Agent Systems (MAS) has deserved an increasing interest in the Intelligent Transportation Systems (ITS) domain. One ITS area of MAS development has been Urban Traffic Control (UTC) systems. In 2000, Ou [6] presented a UTC, which adopted MAS technology based on recursive modeling method (RMM) and Bayesian learning. Ferreira et al. [7] presented a multi-agent decentralized strategy where each agent was in charge of managing the signals of an intersection and optimized an index based on its local state and "opinions" coming from adjacent agents.

In the Advanced Transportation Information System (ATIS) field, Kase and Hattori [8] proposed the InfoMirror application that provides agent-based information assistance to drivers through car navigation systems or on-board PCs. Adorni [9] presented a distributing route guidance system, which allowed dynamic route searching using the coordination capabilities of MAS. Bus-holding control tackles the coordination of multiple lines of fixed-route buses and the different stops, seeking the global optimality. In 2001, Jiamin et al. [10] proposed a distributed bus-holding control approach in which a MAS negotiation between a Bus Agent and a Stop Agent was conducted based on marginal cost calculations.

3 Flexible Public Transport Services

Demand Responsive Transport (DRT) services aim to meet the needs of different users for additional transport supply. The use of flexible transport services, where routes, departure times, vehicles and even operators, can be matched to the identified demand allows a more user-oriented and cost effective approach to service provision. The adaptation of the transport services to match actual demand enables cost savings to the operators, society and passengers.

DRT can be seen as an element of a larger intermodal service chain, providing local mobility and complementary to other conventional forms of transportation (e.g. regular buses and trams, regional trains). In this context, DRT provides a range of Intermediate Transport solutions, filling the gap between traditional public bus services and individual taxis.

The final DRT service can be offered through a range of vehicles including regular service bus, mini-bus, maxi-vans, buses and vans adapted for special needs and regular cars. The use of each vehicle type depends on the transport service to offer, the covered area and the target users.

4 PASSI Methodology

The Process for Agent Societies Specification and Implementation (PASSI) is a step-by-step methodology for designing and developing multi-agent societies.

PASSI integrates design models and concepts from both OO software engineering and artificial intelligence approaches using the UML notation. The design process with PASSI is supported by PTK (PASSI Toolkit [16]) to be used as an add-in for Rational Rose.

The PASSI methodology is made up of five models containing twelve steps in the process of building multi-agent. The models are: System Requirements Model, Agent Society Model, Agent Implementation Model, Code Model and Deployment Model. Because of space restrictions, the present work will focus in the first model. Please refer to [2] for a more detailed description on the whole PASSI methodology.

The System Requirements Model corresponds to an anthropomorphic model of the system requirements in terms of agency and purpose. It involves 4 steps: 1) a Domain Description (D.D.), which provides a functional description of the system using conventional use-case diagrams, 2) an Agent Identification (A.Id.), leveraging the separation of responsibility concerns into agents, represented as UML packages, 3) a Role Identification (R.Id.), consisting in use of sequence diagrams to explore each agent's responsibilities through role-specific scenarios and 4) a Task Specification (T.Sp.), detailing through activity diagrams the capabilities of each agent.

5 The Agent-Based Transportation System

As stated before, the agent system was designed following the PASSI methodology, making use the PTK (Passi Toolkit) add-on for Rational Rose to develop the different models. The system prototype was implemented over the Jade Agent Platform[11], which provides a full environment for agents to work. In the following the general architecture will be explained for then detailing the interface agents in the next sections. For a more detailed description on the agent architecture and the planning & scheduling mechanism please refer to [5] and [3].

As outlined in the PASSI section, the methodology starts capturing the system's requirements through use cases, for then grouping them together to conform the agents. The diagram in Figure 1 shows part of the use cases and agents involved in the system. Due to space restrictions some of the supporting agents are expressed as actors.

The Client is an interface agent with a GUI. Providing the connection between the end user (Customer) and the transportation system. Through it, the Customer can request a trip by giving a description of the desired transportation service through a *Trip Request Profile*. In addition, through a *Client Profile* it is possible to create and manage personalized services profiles with diverse characteristics and preferences common to the different trips requested by the user.

After a service has been contracted, the Customer can also communicate events to the system (e.g. a delay, a change on the agreed service, or simply cancel). In a similar way, the system can communicate with the Customer, informing him about any eventuality that may happen (e.g. a traffic jam or vehicle break

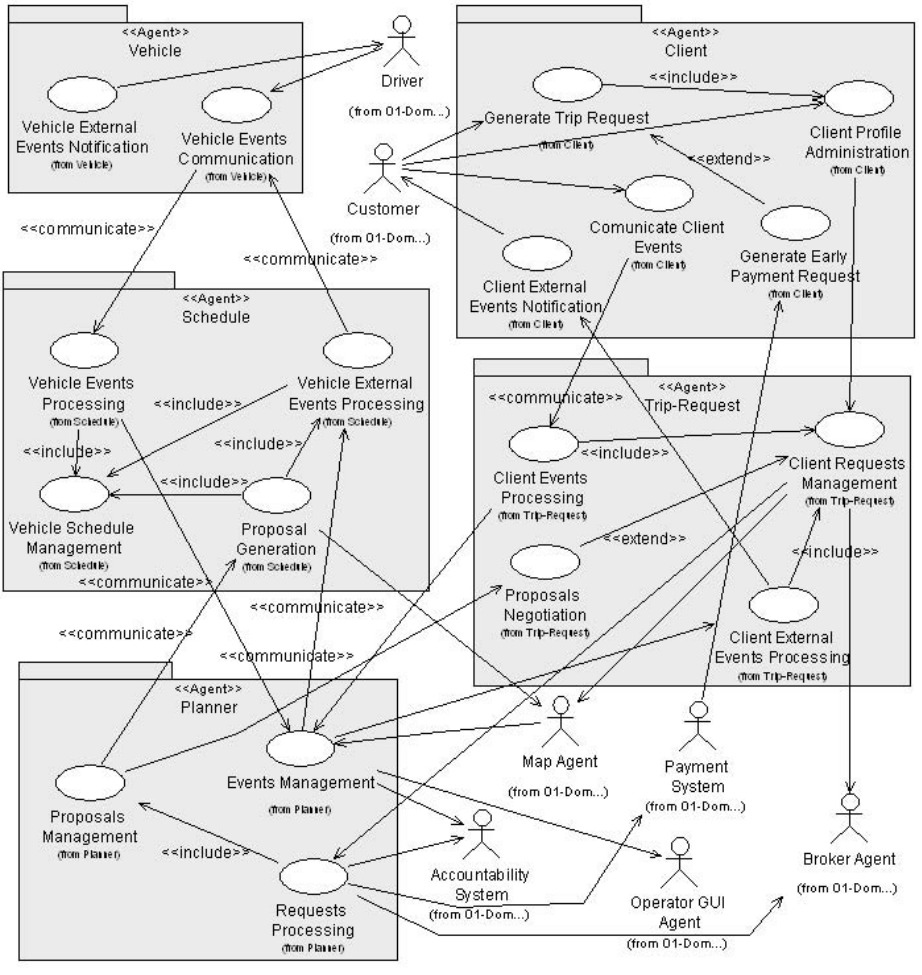


Fig. 1. Portion of the Agents' Identification Diagram

down) which may imply a delay or change in the service to be provided. This agent will be further detailed in the next section. The Trip-Request agent acts as a proxy, representing the Customer in the process of contracting a transportation service. In fact, the trip-request agent is involved in all the interaction of the Customer (through the interface agent) with the transportation system. Its activities regard the management of the client transportation requests, including any negotiation or selection of proposals coming from the Planner, together with processing any events generated by the Customer or by the system. As residing on a device with more processing power (such as a PC), this agent may have diverse degrees of autonomy for taking decisions on the trip proposal to choose and how to react when faced to eventualities.

The Vehicle is also an interface agent (with a GUI) in charge of providing a monitoring of the route-schedule planned for the vehicle. In addition, it can inform the Driver about any changes to the initial schedule and can be used by him to inform any eventuality (e.g client no show, delay, detour, etc) that may happen regarding the trip and the customers. In particular its interface has been designed to work on-board the vehicle through a touch screen.

The Schedule agent is the one in charge of managing the trip plan (work-schedule) of the vehicle. In addition, the agent is also responsible of making trip proposals upon Planner request and in case of winning will have to update its actual plan to include the new trip. Upon changes (due to vehicle or client events) informed either by the Vehicle or the Planner agent, the Schedule agent will update the plan and reschedule the remaining requests.

Finally, the Planner agent processes all the customers' requests coming through their Trip-request agents. It initiates a contract-net (CNP) [17] with the Schedule agents and manages all the arrived proposals. It is also in charge of managing events that may affect the trip services already contracted and scheduled.

The rest of the actors correspond to supporting-service agents or systems that interact with the diverse agents already detailed, such as the broker, responsible for the initial service matching, the map, providing times and distances, and the payment, responsible for a secure and reliable payment transaction, among others.

5.1 The Agent Interaction

In PASSI the agent interaction is modeled through sequence diagrams that show the diverse scenarios in which agents communicate. The following Figure 2 shows part of the scenario in which a Customer requests a trip service. Each object in the diagram is described following the $\langle role \rangle : \langle agent \rangle$ convention. Therefore, this scenario involves the Customer, Map and Broker actors plus the Client, Trip-request, Planner and Schedule agents.

The scenario starts with the Customer initiating the request of a trip through the client interface by clicking in the top menu. The Customer fills-in the information requested in the form and the interface stores a corresponding *Request Profile*. Then, the *Trip Request Generator* role of the Client generates the Trip-request agent providing the *Request Profile* as argument. This agent initiates the request with the Planner through its *Client Requests Manager* role. The Planner receives the request and queries the Broker for registered vehicles fulfilling the *Request Profile*. The Broker returns a list of possible vehicles and the Planner starts analyzing the alternatives through its *Proposals Manager* role. In fact, this role performs the Manager activities of the Contract-Net protocol. Then, it initiates a call for proposals to the corresponding Schedule agents according to the vehicles' list.

The Schedule agent encapsulates the underlying optimization algorithm for scheduling the trips of the vehicle. In our implementation Schedule agents implement a distributed version of a well known greedy insertion algorithm called ADARTW (Please refer to [3] for further details).

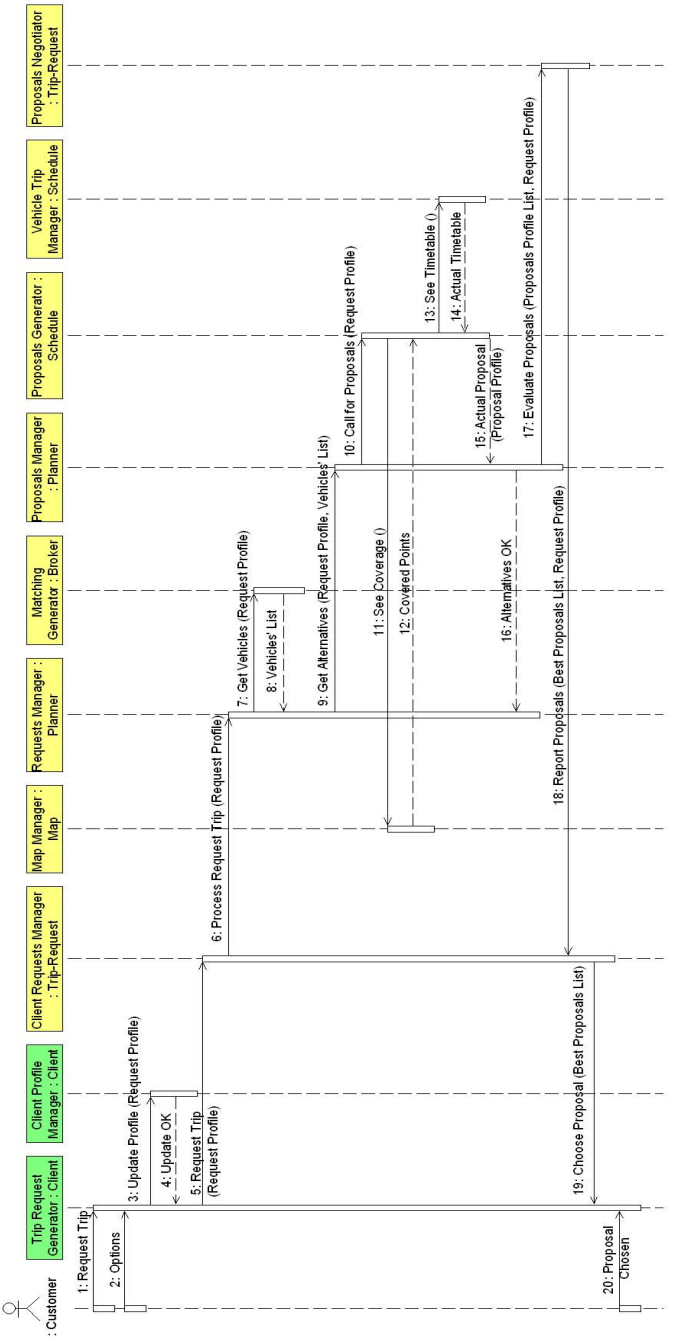


Fig. 2. Roles Identification: Part of the "Customer Requests a Trip Service" scenario

The *Proposals Generator* role of the Schedule agent analyzes the *Request Profile*, first by requesting the Map for the coordinates of the pickup and delivery addresses and some paths and distances to evaluate incorporation alternatives to the actual schedule of the vehicle. The Schedule agent turns back a valid proposal or a refuse performative. The Planner waits for proposals until a deadline or until receiving all answers back, asking the Trip-request to evaluate the proposals. The *Proposals Negotiator* role of the Trip-request will process the alternatives and depending on its degree of autonomy, can decide on behalf of the Customer or can report the list of proposals to the Client agent for the Customer to choose.

6 The Client Agent

As stated before, the Client is an interface agent devoted to the Customer-System interaction. In principle, this trip-client assistant may reside on diverse devices (e.g PC, PDA, mobile phone) in order to allow a more flexible and pervasive access to the transportation system. In our prototype, has been developed a Client agent for PC, remaining the versions for more restricted devices as future work. In this sense, it is important to highlight that all the complex processing or decision-making (if delegated by the Customer) has been attached to the Trip-request agent in order to lightweight the Client (the interface agent).

In the following Figure 3 a screenshot of the Client agent GUI is shown, detailing the tab that appears when initiating the request of a trip. In the "Request Data" area, on the left, is asked all the information necessary to detail a transport service request under the demand-responsive considered scenario. This regards the date, the pickup and delivery points (addresses), the corresponding times and other specific information such as the required seats and diverse vehicle characteristics.

It is important to mention that all the concepts involved in the specification of the services make part of a Domain Ontology specific for this transportation domain (for further details on the ontology please refer to [5]).

On the right hand, the available transport services are deployed, showing for each selected service the covered area in terms of street intersections. The services' list can be imported from the system (on line) or from a local file. At the bottom, the Customer can send the trip request and save the services' list.

The PASSI methodology used for the modeling considers a *Task Specification* step. In this activity the scope is to focus on each agents behavior, decomposing it into tasks which usually capture some functionality that conforms a logical unit of work. Therefore for each agent an activity diagram is developed, containing what that agent is capable of along the diverse roles it performs. In general terms, an agent will be requiring one task for handling each incoming and outgoing message.

In the following Figure 4 a portion of the Task Specification Model for the Client Agent is depicted. The diagram shows six tasks that constitute the main Client agent capabilities devoted to the process of requesting a transportation service. The *SendQueryAvailableService* task handles the request from the

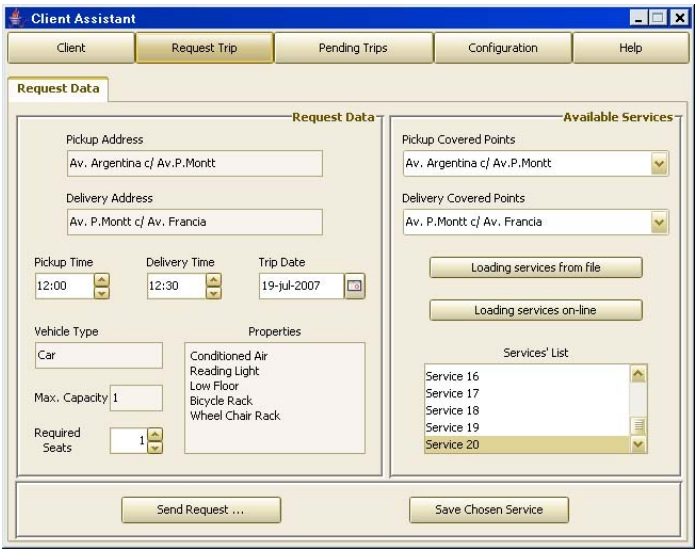


Fig. 3. Client agent GUI showing the "Request Data" tab in the "Request Trip" menu

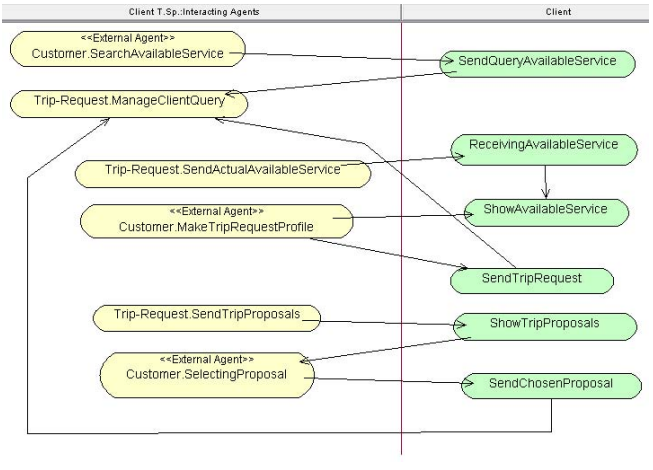


Fig. 4. Part of the Task Specification for the Client Agent, showing the flow of tasks involved in the trip request processing

Customer to search for available services and triggers the *ManageClientQuery* task of the Trip-Request agent which is in charge of requesting the Broker for possible transportations services available. These are returned by the *SendActualAvailableService* task of the Trip-request and is received by the *ReceivingAvailableService* task of the Client which processes and decodes the ACL message and forwards the services' list to the *ShowAvailableService* task responsible for displaying the list in the proper form.

The Customer, when making a trip Request Profile (see Figure 3), can browse on the available services (after loading them) in the right-hand area calling to the *ShowAvailableService* task or can send the request (by pressing the button) after filling the left-hand information, calling the *SendTripRequest* task. This Client's task is responsible for sending the *Request Profile* to the Trip-Request, being handled by its *ManageClientQuery* task, which on its turn will forward the request to the Planner.

As explained in the "Customer Requests a Trip Service" scenario of section 5.1, the Trip-request agent will receive from the Planner the trip proposals coming from the different vehicles' Schedule agents and its *SendTripProposals* task will send them to the Client. On its turn, the Client will receive and handle the proposals through its *ShowTripProposals* task, also responsible for displaying them on an appropriate form. Finally, the Customer will be able to select the best alternative calling to the *SendChosenProposal* task of the Client.

7 Conclusions

The design of an agent system devoted to passenger transportation under a demand-responsive approach was described. The PASSI methodology used allowed an appropriate level of specification along its diverse phases. The present work focused in the specification of the interface agents for the main actors involved with the system: Customers and Drivers. The Client interface Agent prototype was detailed providing an in-depth example of agent design & implementation using the AOSE PASSI. Future work considers enable the system openness with the implementation of a high-level communication mechanism in order to provide a dynamic participation in such a system.

Acknowledgement

This work is part of Project No. 209.746/2007 entitled "Coordinación en una sociedad multiagente dedicada a la programación y control bajo ambiente dinámico", funded by the Pontifical Catholic University of Valparaíso (www.pucv.cl).

References

1. Weiss, G.: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, Massachusetts, USA (1999)
2. Burrafato, P., Cossentino, M.: Designing a multiagent solution for a bookstore with the passi methodology. In: 4th International Bi-Conference Workshop on AgentOriented Information Systems (AOIS-2002) (2002)
3. Cubillos, C., Crawford, B., Rodríguez, N.: Distributed Planning for the On-Line Dial-a-Ride Problem. In: Preparata, F.P., Fang, Q. (eds.) FAW 2007. LNCS, vol. 4613, pp. 124–135. Springer, Heidelberg (2007)
4. Cubillos, C., Rodríguez, N., Crawford, B.: A Study on Genetic Algorithms for the DARP Problem. In: Mira, J., Alvarez, J.R. (eds.) IWINAC 2007, Part I. LNCS, vol. 4527, pp. 498–507. Springer, Heidelberg (2007)

5. Cubillos, C., Gaete, S.: Design of an Agent-Based System for Passenger Transportation using PASSI. In: Mira, J., Alvarez, J.R. (eds.) IWINAC 2007, Part II. LNCS, vol. 4528, pp. 531–540. Springer, Heidelberg (2007)
6. Ou, H.T.: Urban Traffic Multi-Agent System based on RMM and Bayesian Learning. In: Proc. American Control Conference 2000, pp. 2782–2783 (2000)
7. Ferreira, E.D., Subrahmanian, E.: Intelligent Agents in Decentralized Traffic Control. In: IEEE Intelligent Transportation Systems Conference Proceedings, USA, pp. 705–709 (August 2001)
8. Kase, N., Hattori, M.: InfoMirror - Agent-based Information Assistance to Drivers. In: IEEE\IEEJ\JSAI Intelligent Transportation Systems Conference Proceedings, pp. 734–739 (1999)
9. Adorni, G.: Route Guidance as a Just-In-Time Multiagent Task. Journal of Applied Artificial Intelligence 10(2), 95–120 (1996)
10. Jiamin, Z., Dessouky, M., Bukkapatnam, S.: Distributed Holding Control of Bus Transit Operations. In: IEEE Intelligent Transportation Systems Conference Proceedings, Oakland - USA, pp. 976–981 (August 2001)
11. Bellifemine, F., et al.: JADE: - A FIPA Compliant Agent Framework. C SELT Internal Technical Report (1999)
12. Bürckert, H., Fischer, K., et al.: TeleTruck: A Holonic Fleet Management System. In: 14th European Meeting on Cybernetics and Systems Research, pp. 695–700 (1998)
13. Fischer, K., Müller, J.P., Pischel, M.: Cooperative Transportation Scheduling: An application Domain for DAI. Journal of Applied Artificial Intelligence 10 (1996)
14. Kohout, R., Erol, K., Robert, C.: In-Time Agent-Based Vehicle Routing with a Stochastic Improvement Heuristic. In: Proc. Of the AAAI/IAAI Int. Conf. Orlando, Florida, pp. 864–869 (1999)
15. Perugini, D., Lambert, D., et al.: A distributed agent approach to global transportation scheduling. In: IEEE/ WIC Int. Conf. on Intelligent Agent Technology, pp. 18–24 (2003)
16. PASSI Toolkit (PTK) Available at http://sourceforge.net/projects/ptk
17. Smith, R.G., Davis, R.: Distributed Problem Solving: The Contract Net Approach. In: Proceedings of the 2nd National Conference of the Canadian Society for Computational Studies of Intelligence (1978)

A Multi-agent System Approach to Power System Topology Verification

Kazimierz Wilkosz

Wroclaw University of Technology, Wybrzeze Wyspianskiego 27,
50-370 Wroclaw, Poland
Kazimierz.Wilkosz@pwr.wroc.pl

Abstract. The paper deals with power system topology verification, being an important problem in real-time power system modeling. In the paper, the approach with use of multi-agent system is proposed. At the beginning, benefits from utilization of the agent technology are presented. Then, a theoretical background for the power system topology verification is described. Next, multi-agent system for topology verification and functions of particular agents are characterized. At the end, features of the presented approach to power system topology verification are summed up.

Keywords: Power System, Power System Topology, Multi-Agent System.

1 Introduction

Advances in computer technology (in the field of the hardware and software), telecommunication, and electric power engineering give new possibilities for control and management of power systems. On the other hand the power system control in a more and more demanding environment becomes more and more complex. The required attributes of the modern control and management of power systems are scalability, openness, flexibility and conformance to industry standards. Now, achievement of the solution, which has such attributes, seems to be easier, using (among other things) intelligent agents and multi-agent systems. The advantages of the Agent Technology (AT) from the view point of their utilization for solving problems of the mentioned control and management are underlined in many papers [1] - [5]. The following features of AT are especially interesting:

- AT can be effectively used to solve complex (distributed) problems,
- agents are loosely coupled,
- agents can communicate via messaging rather than by procedure calls,
- new functions can easily be added to an agent-based system by creating a new agent, which will then make its capabilities available to others,
- agents can be stopped without affecting the integrity of the other agents in the system
- AT permits the easy integration of data capture, data processing, and intelligent system interpretation processes,

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 970–979, 2007.

- the management of data, information and interpretation can be devolved to individual agents, permitting dynamic and automatic decision making regarding which interpretation algorithms should be used,
- the data interpretation functions can be distributed, removing the need for wholesale data transmission,
- agents can ultimately reduce operator errors and improve control performance by responding to problems faster than a human operator.

AT is one of the recent developments in the field of distributed artificial intelligence. There is relatively small number of papers showing application of this technology to solving problems related to real-time power system modeling (power-system state estimation) [1], [2], [6], [7]. One of such problems is the problem of power system Topology Errors (TEs) and the problem of the Power System Topology Verification (PSTV). PSTV, i.e. proving or disproving the correctness of a power system topology model, is an important problem when a real-time power system model is built. The topology model, being a description of the physical connections in a power system, is essential part of a power system model. Only in [6] the problem of PSTV is considered.

The focus of the paper is utilization of AT for PSTV from the functional viewpoint. In the paper, other than in [6], an essential element of approach for PSTV is utilization of knowledge on a topology model which is contained in the relationships among measured quantities in a power system. Utilization of AT permits realization of a distributed PSTV. Results of such PSTV can be accessed locally by different applications.

2 The Theoretical Background of the Approach

2.1 A Power System Model

The considered approach assumes utilization of a bus/node power system model. In the model, nodes and branches are distinguished. The model nodes represent electrical nodes in a power system. The model branches represent connections between suitable electrical nodes in a power system, i.e. power lines, transformers etc.

The assumption is made that every branch in a power network is modeled as the π-equivalent circuit (Fig. 1). It is assumed that there is an accessible credible measurement data set of such quantities as active and reactive power flows at the ends of each branch, power injections, loads and voltage magnitudes at each node.

2.2 Utilized Relationships

A power system is described by:

- relationships among active and also reactive power flows for nodes:

$$\sum_{i \in I_k} P_{ki} = 0\,, \quad \sum_{i \in I_k} Q_{ki} = 0\,, \tag{1}$$

where: k – a number of the considered node; i – a number of the node which is connected with the node k by a branch; I_k – the set of nodes connected with the

node k; P_{ki}, Q_{ki} – respectively active and reactive power flows at the node k in the branch connecting the nodes k and i;

– relationships among active and reactive power flows and also voltage magnitudes at the ends of branches:

$$P_{kl} + P_{lk} + R_{kl} \frac{P_{kl}^2 + \left(Q_{kl} - B_{kl} V_k^2\right)^2}{V_k^2} = 0, \tag{2}$$

$$Q_{kl} + Q_{lk} + X_{kl} \frac{P_{kl}^2 + \left(Q_{kl} - B_{kl} V_k^2\right)^2}{V_k^2} - B_{kl}\left(V_k^2 + V_l^2\right) = 0. \tag{3}$$

where: k, l –numbers of the nodes between which there is the considered branch; V_k, V_l –voltage magnitudes at the nodes k and l respectively; R_{kl}, X_{kl}, B_{kl} – parameters of the π model of the branch**.**

The relationships **(1)** result from the Kirchhoff's Current Law. The relationships (**2**), (**3**) are a result of applications of the Kirchhoff's and Ohm's Laws for a branch.

2.3 A Classical Approach to Power System Analyses

If a node or a branch is not included in a power system model, then the appropriate measurement data related to this node or to this branch are not taken into account. In such a case the relationships **(1)** for the mentioned node or the relationships (**2**), (**3**) for the mentioned branch are not considered.

2.4 The Utilized Idea of Power System Analyses

In this paper, relationships for all possible nodes and all possible connections among these nodes are considered in all possible cases of network connectivity.

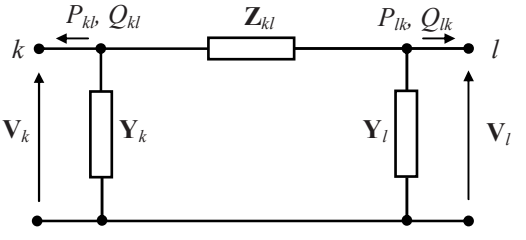


Fig. 1. The assumed π model of the branch. $\mathbf{Z}_{kl} = R_{kl} + \mathrm{j}\, X_{kl}$, $\mathbf{Y}_k = \mathrm{j}B_{kl}$, $\mathbf{Y}_l = \mathrm{j}B_{lk}$, $B_{kl} = B_{lk} = B$. B is a half of the capacitive susceptance of the branch.

If a node is not included in the power system model, then for testing the relationships **(1)** the assumption is made that all active and reactive power flows at the considered node are equal to zero. If the considered node occurs in a power system and also occurs in the topology model of the power system then there are taken into

account measurement data of active and reactive power flows at the ends of the branches which enter or which could enter the node.

If the branch between the nodes k and l (the branch k-l) is not included in the power system model then for the terminal nodes of the branch

$$W_{Pk} = -P_{kl}, \quad W_{Pl} = -P_{lk}, \; W_{Qk} = -Q_{kl}, \quad W_{Ql} = -Q_{lk}, \tag{4}$$

where

$$W_{Px} = \sum_{i \in I_x} P_{xi}, \; W_{Qx} = \sum_{i \in I_x} Q_{xi}, \tag{5}$$

$x \in \{k, l\}$.

For the considered branch, substituting in the relationships (**2**), (**3**) the power flows P_{kl}, P_{lk}, Q_{kl}, Q_{lk} with the values $-W_{Pk}$, $-W_{Pl}$, $-W_{Qk}$, $-W_{Ql}$ respectively, we have

$$W_{Pkl} = -W_{Pk} - W_{Pl} + R_{kl} \frac{W_{Pk}^2 + \left(W_{Qk} + B_{kl} V_k^2\right)^2}{V_k^2}, \tag{6}$$

$$W_{Qkl} = -W_{Qk} - W_{Ql} + X_{kl} \frac{W_{Pk}^2 + \left(W_{Qk} + B_{kl} V_k^2\right)^2}{V_k^2} - B_{kl}\left(V_k^2 + V_l^2\right). \tag{7}$$

If in a real power system the branch k-l is in operation, but it is not included in the topology model i.e. the branch is incorrectly modeled (the exclusion error), and power flows in this branch are not equal to zero then

$$W_{Pk} \neq 0, \quad W_{Pl} \neq 0, \; W_{Qk} \neq 0, \quad W_{Ql} \neq 0, \; W_{Pkl} = 0, \; W_{Qkl} = 0. \tag{8}$$

If in a real power system the branch k-l is out of operation and it is included in the topology model (the branch is incorrectly modeled – the inclusion error) or the considered branch is correctly modeled then

$$W_{Pk} = 0, \quad W_{Pl} = 0, \; W_{Qk} = 0, \quad W_{Ql} = 0, \tag{9}$$

$$W_{Pkl} = R_{kl} B_{kl}^2 V_k^2, \; W_{Qkl} = X_{kl} B_{kl}^2 V_k^2 - B_{kl}\left(V_k^2 + V_l^2\right). \tag{10}$$

In the assumed approach the so-called unbalance indices for nodes and branches are introduced. The idea of such indices is described in [8]. In the presented approach, for nodes these indices (the indices W_{Px}, W_{Qx} x – a number of a node) are defined as in [8], i.e. with use of the formulae (**5**), for branches - with use of the following formulae:

$$W_{Pkl} = \begin{cases} -W_{Pk} - W_{Pl} + R_{kl} W & \text{if} \quad W_{Px} \neq 0 \text{ and } W_{Qx} \neq 0 \\ -W_{Pk} - W_{Pl} + R_{kl} W^0 & \text{if} \quad W_{Px} = 0 \text{ and } W_{Qx} = 0 \end{cases}, \tag{11}$$

$$W_{Qkl} = \begin{cases} -W_{Qk} - W_{Ql} + X_{kl} W - B_{kl}\left(V_k^2 + V_l^2\right) & \text{if} \quad W_{Px} \neq 0 \text{ and } W_{Qx} \neq 0 \\ -W_{Qk} - W_{Ql} + X_{kl} W^0 & \text{if} \quad W_{Px} = 0 \text{ and } W_{Qx} = 0 \end{cases}. \tag{12}$$

where:

$$W = \frac{W_{Pk}^2 + \left(W_{Qk} + B_{kl} V_k^2\right)^2}{V_k^2}, \; W^0 = \frac{W_{Pk}^2 + W_{Qk}^2}{V_k^2}. \tag{13}$$

Using the mentioned indices, one can investigate the relationships among the measured quantities for branches independently of their inclusion into the power system topology model or exclusion from this model. It should be stressed that the behavior of unbalance indices for active power and for reactive power is the same for the same TE.

Unbalance indices create characteristic sets of values for different cases of modeling a power system. If the topology model is correct and there are no errors burdening measurement data then all the unbalance indices for nodes and branches are equal to zero. The same situation is in a case of the inclusion error.

The case in which:

- the unbalance indices for the distinguished node *k* and for each node connected with the node *k* by a branch are equal to zero,
- the unbalance indices for each branch incident to the node *k* are equal to zero,

is further called *Case* 0.

In a case of the exclusion error one can observe:

- the unbalance indices for terminal nodes of the branch, which is excluded from the topology model, are considerably different from zero,
- the unbalance indices for the considered branch (i.e. the branch *k*-*l*) are equal to zero,
- absolute values of the unbalance indices for each branch, being incident to the mentioned terminal nodes and not being the branch *k*-*l*, are especially large,
- values of the unbalance indices for each node, which is connected with the node *k* or with the node *l* by a branch (other than branch *k*-*l*), are equal to zero.

The case in which:

- the unbalance indices for the distinguished node *k* and for the node *l*, which in a power network can be connected with the node *k* by a branch (i.e. by the branch *k*-*l*), are considerably different from zero,
- the unbalance indices for the branch *k*-*l* are equal to zero,
- absolute values of the unbalance indices for each branch, that is incident to the node *k* and is not the branch *k*-*l*, have especially large values,
- absolute values of the unbalance indices for each node, which is connected with the node *k* by a branch and is not the node *l*, are equal to zero,

is further called *Case* 1.

Analyzing unbalance indices for nodes and branches one can observe that the exclusion error of the branch *k*-*l* has no influence on:

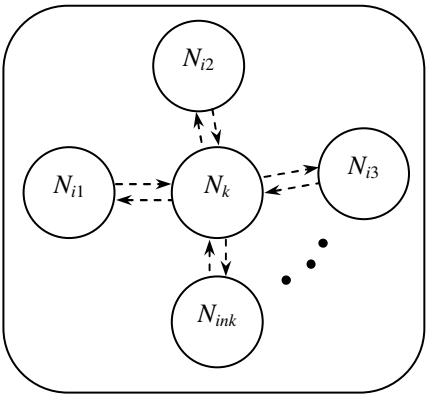


Fig. 2. Co-operation among the node agents

– unbalance indices for nodes, that are not terminal nodes of the branch *k-l*,
– unbalance indices for branches, that are not incident to terminal nodes of the branch *k-l*.

This observation shows existence of the local effect of TE. In this situation one can conclude about correctness of modeling the distinguished branch *k-l* on the basis of investigations of unbalance indices for certain areas of the power network: A^k_{k-l}, A^l_{k-l}.

The area A^x_{k-l} $x \in \{k, l\}$ comprises:

– the node *x*,
– the branch *k-l* and all other branches incident to the node *x*,
– all nodes which are connected with the node *x* by the earlier-mentioned branches.

The analysis of unbalance indices is relatively simple in the case of a single TE and when impact of errors burdening measurement data is not considered. In real cases, existence of multiple TEs and errors burdening measurement data should be taken into account. In such cases, the problem of PSTV is a complex problem. PSTV can be treated as a problem of pattern recognition and then ANNs can be utilized for solution of this problem, e.g. as it was presented in [9].

3 A Description of the Approach

Process of PSTV consists of many processes, which are not independent. Each distinguished process is performed by an intelligent agent. There are two types of agents. There are agents related to nodes (the node agents) and agents related to branches (the branch agents). The agents related to nodes are denoted by

N_j $j = 1, 2, ..., n$, where j – the number of a node, and the agents related to branches are denoted by $B_{j\text{-}i}$ $i, j = 1, 2, ..., n$ $i \neq j$, where i, j – the numbers of terminal nodes of a branch. Relationships among agents are shown in Fig. 2 and Fig. 3. The agent N_k $k \in \{1, 2, ..., n\}$ co-operates with:

– each agent N_{i1} $i_1 \in I_k$, i.e. with each agent related to the node which is connected or can be connected (when in the power system topology model there is not a connection) with the node N_k by a branch,
– each agent $B_{k\text{-}i1}$ $i_1 \in I_k$, i.e. with each agent related to the branch which connects or can connects the node k with the node i_1.

3.1 The Node Agents

Each of the node agents realizes functions: F_{N1}, F_{N2}, F_{N3}, F_{N4} (Fig. 4). It observes its environment (a substation), more exactly, measurement data of active and reactive

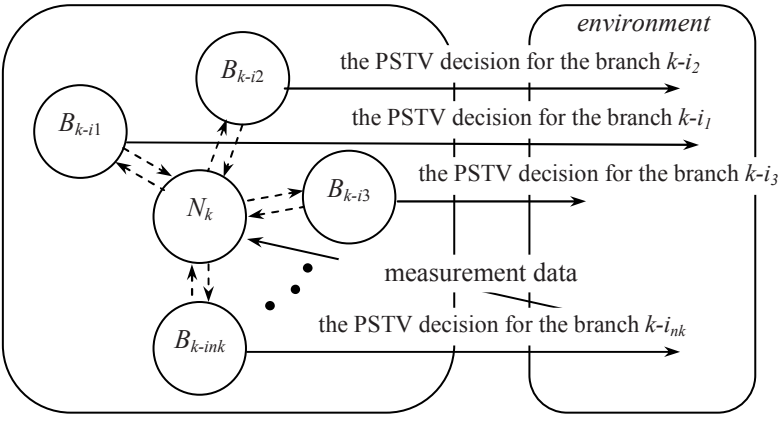


Fig. 3. Co-operation between the node agent N_k and the branch agents $B_{k\text{-}x}$ $x = i_1, i_2, ..., i_{nk}$

power flows at the ends of branches and voltage magnitudes at the buses. When there is any change of measurement data the appropriate node agent starts its action. The aim of this action is to take local decisions on correctness of modeling of branches entering the node with which the node agent is related.

The possible decisions are as follows: D_c – the branch is correctly modeled, D_b – the branch is incorrectly modeled (there is TE), D_0 – there is no basis for taking decision D_c or D_b. Information on these decisions is sent to the appropriate branch agents.

Information is exchanged among node agents with use of data sets. These are:

DS_{Nkx} – the data set created by the node agent N_k and addressed to the node agent N_x. The set contains the unbalance indices: W_{Pk}, W_{Qk} and the data informing the node agent N_x about a request of sending the unbalance indices: W_{Px}, W_{Qx} to the node agent N_k.

DS_{NkxR} - the data set, created by the node agent N_x, with the unbalance indices: W_{Px}, W_{Qx} and addressed to the node agent N_k. The data set is an answer for the request contained in the data set DS_{Nkx}.

$DS_{NkBk\text{-}x}$, $DS_{NxBk\text{-}x}$ – the data sets created by the node agent N_k or by the node agent N_x respectively with the decision on correctness of modeling of the branch $k\text{-}x$, taken by the node agent, and addressed to the branch agent $B_{k\text{-}x}$.

$DS_{Bk\text{-}xNk}$, $DS_{Bk\text{-}xNx}$ - the data sets created by the branch agent $B_{k\text{-}x}$ and addressed to the node agent N_k or to the node agent N_x respectively with the data informing the node agent about a request of sending its decision on correctness of modeling of the branch $k\text{-}x$ to the branch agent $B_{k\text{-}x}$.

$DS_{Bk\text{-}xNkR}$, $DS_{Bk\text{-}xNxR}$ - the data sets created by the node agent N_k or by the node agent N_x respectively with the decision on correctness of modeling of the branch $k\text{-}x$, taken by the node agent, and addressed to the branch agent $B_{k\text{-}x}$. The data sets are answers for the requests contained in the data sets $DS_{Bk\text{-}xNk}$, $DS_{Bk\text{-}xNx}$ respectively.

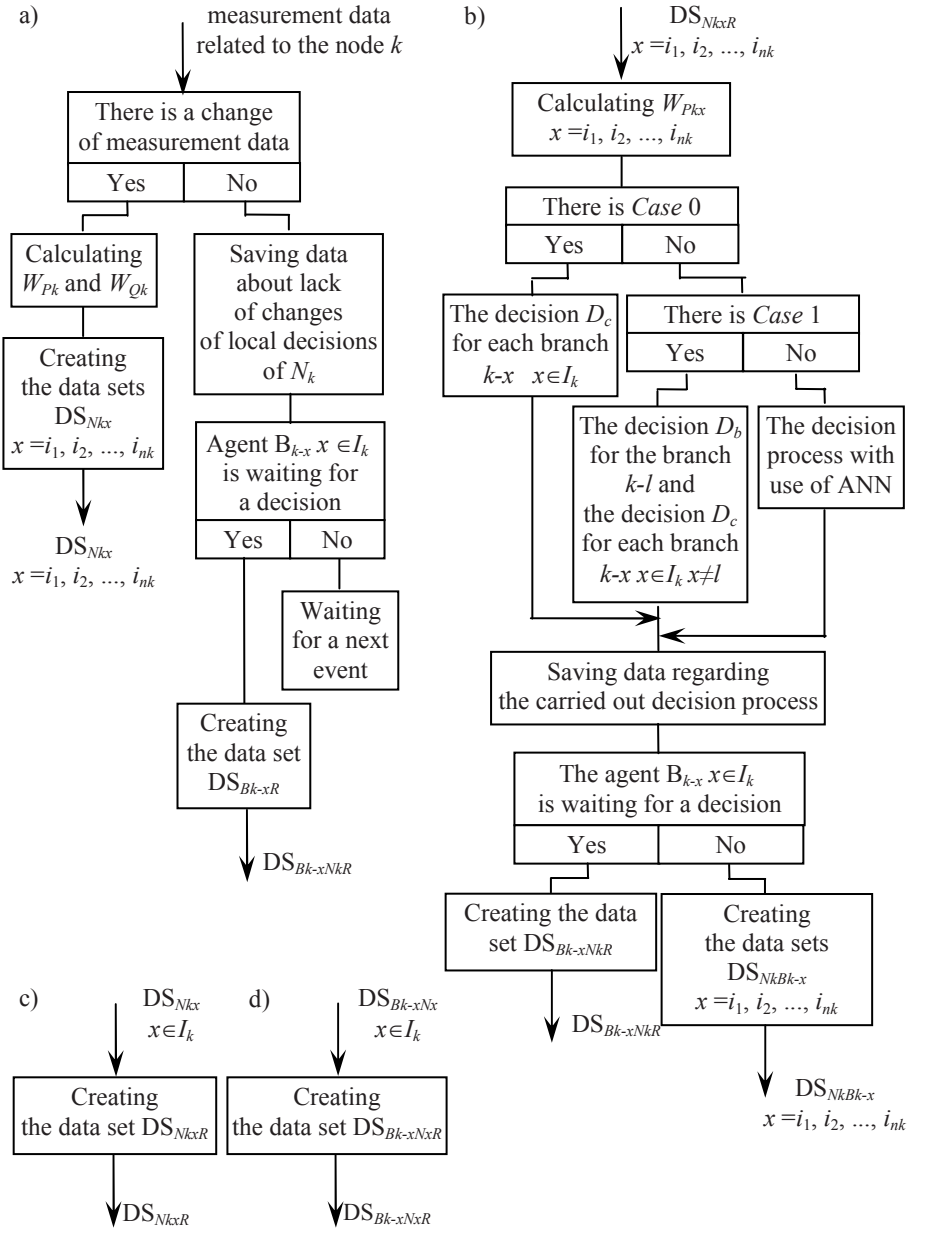


Fig. 4. The functions of the node agent: a) the function F_{N1}, b) the function F_{N2}, c) the function F_{N3}, d) the function F_{N4}. Description of the designations is in Subsection 3.1.

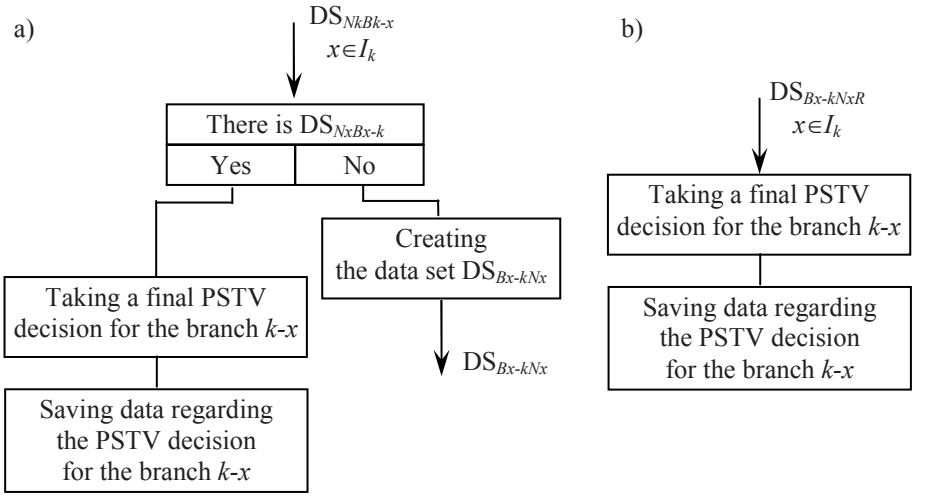


Fig. 5. The functions of the branch agent: a) the function F_{B1}, b) the function F_{B2}. Description of the designations is in Subsection 3.1.

3.2 The Branch Agents

Each of the branch agents realizes functions: F_{B1}, F_{B2} (Fig. 5). The branch agent takes the final PSTV decision for the branch to which it is related. The bases for taking this decision are the decisions taken by both node agents related to the terminal nodes of the considered branch.

4 Conclusions

Utilization of unbalance indices for PSTV gives new possibilities. A process of PSTV can be considered as many processes which are performed locally. For the so-considered PSTV the multi-agent system approach is proposed. According to this approach the intelligent agents for nodes and the intelligent agents for branches of a power network are foreseen. One electrical node is related to one substation. In one substation, one or more electrical nodes can occur. Thus more than one node agent can be related to one substation. A decision on correctness of modeling selected branch is produced as a result of co-operation of the intelligent agent for the considered branch, the intelligent agents for the terminal nodes of the branch and for nodes neighboring to the mentioned terminal nodes.

Analyzing the described utilization of AT in PSTV gives basis for listing, among other things, the following benefits of the presented multi-agent system approach:

1. The problem of PSTV can be considered as a set of many smaller problems.
2. The system for PSTV is easily scalable - the organizational structure of the agents can dynamically change as the power network grows in size.

3. The locally-verified data on connections in a power network can be transmitted to each control centre and in effect a time of real-time modeling of a power system in this control centre can be shorter what is very important from the view point of real-time power system applications utilizing the real-time power system model.

References

1. Biernatzki, R., Bitzer, B., Zivanovic, R., Els, A., Els, S.: Agent Technology Used in Power Systems. In: The 7th AFRICON Conference, Africa, vol. 2, pp. 705–710 (2004)
2. Moslehi, K., Kumar, A.B.R., Dehdashti, E., Hirsch, P., Wu, W.: Distributed Autonomous Real-Time System for Power System Operations-a Conceptual Overview. IEEE PES Power Systems Conference and Exposition 1, 27–34 (2004)
3. McArthur, S.D.J., Davidson, E.M.: Concepts and Approaches in Multi-Agent Systems for Power Applications. In: The 13th International Conference on Intelligent Systems Application to Power Systems, pp. 391–395 (2005)
4. McArthur, S.D.J.: Agent Technologies for Power System Plant Monitoring and Diagnosis. In: The IEE Seminar on Autonomous Agents in Control, pp. 51–65 (2005)
5. de Azevedo, G.P., Feijo, B.: Agents in Power System Control Centers. IEEE Power Engineering Society General Meeting 2, 1040–1041 (2005)
6. Nordman, M.M., Lehtonen, M.: Distributed Agent-Based State Estimation for Electrical Distribution Networks. IEEE Trans. on Power Systems 2, 652–658 (2005)
7. Qiong, W., Wenyin, L., Yihan, Y., Chuan, Z., Yong, L.: Intelligent Decision Support System for Power Grid Dispatching Based on Multi-Agent System. In: International Conference on Power System Technology (PowerCon), pp. 1–5 (2006)
8. Lukomski, R.: New Approach to Power System Topology Verification. In: The International Conference on Modern Electric Power Systems, pp. 517–521. Wroclaw, Poland (2002)
9. Lukomski, R., Wilkosz, K.: Method for Power System Topology Verification with use of Radial Basis Function Networks. In: Sandoval, F., Prieto, A., Cabestany, J., Graña, M. (eds.) Computational and Ambient Intelligence. LNCS, vol. 4507, pp. 862–869. Springer, Heidelberg (2007)

A System for Efficient Portfolio Management

Vivian F. López[1], Luis Alonso[1], María N. Moreno[1],
Saddys Segrera[1], and Alfredo Belloso[2]

[1] Dept. Informática y Automática. University of Salamanca,
Plaza de la Merced S/N, 37008. Salamanca
{vivian,lalonso,mmg, saddys}@usal.es
http://dptoia.usal.es
[2] Dept. Económico y Financiero. ADIF.
Avenida de Burgos. Madrid
abelloso@adif.es

Abstract. In this work we perform an automatic data survey to draw up an optimum portfolio, and to automate the one year forecast of a portfolio's payoff and risk, showing the advantages of using formally grounded models in portfolio management and adopting a strategy that ensures, a high rate of return at a minimum risk. The use of neural networks provides an interesting alternative to the statistical classifier. We can take a decision on the purchase or sale of a given asset, using a neural network to classify the process into three decisions: buy, sell or do nothing.

Keywords: portfolio payoff risk neural network.

1 Introduction

The decision-making process about when, how and what to invest is one of the most evocative issues in the investment world. These decisions challenge the investor's entire range of knowledge, which is always complicated, but particularly nowadays, when the exchange markets are highly volatile. Such is the case of portfolio management, which is still performed as craftwork. Selection is made according to the investor's favourite assets, or following the manager's ratings according to her experience, knowledge or intuition, but seldom based on formal grounds. This means that investors maintain inefficient portfolios that are not adjusted to the expected risk-payoff ratio.

Currently, portfolio analysis can be approached from two points of view. First, we have portfolio selection, which Harry Markowitz introduced in 1952 [9]. The second aspect is portfolio management aimed at finding the optimal structure. Today, financial market problems are often solved using artificial intelligence. Despite the great deal of effort already put into making financial time series predictions [8], support vector machines [5], neural networks [11], prediction rules [3] and genetic algorithms [1], the prediction of a stock market index is still difficult to attain. The main reason for the complexity of this task is the lack of autocorrelation of index value which changes even in a one-day period. The aim

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 980–989, 2007.

of this study is to explore once again the application of neural networks to the portfolio management problem.

According to Markowitz, the selection is grounded in the simple observation of prices that maximize the expected payoff at a given level of risk. Although the information is growing day by day, its in-depth processing is very complicated and not within easy reach of the average investor, who is usually unable to capture and interpret the data. In this work we perform an automatic data survey to draw up an optimum portfolio, to estimate the market risk and, at a given moment, to help the decision process regarding an asset. The main objectives of the system are:

1. To automate the one year forecast of a portfolio's payoff and risk, showing the advantages of using theoretically grounded models in portfolio management and adopting a strategy that ensures a high rate of return at minimum risk.
2. To make the correct decision in the purchase or sale of a given asset, using a neural network to classify the process into three decisions: buy, sell or do nothing.

1.1 Portfolio Theory

Markowitz established the aim of setting up the menu of the possible payoff-risk combinations that are eligible, giving as the decision variable the weight or ratio assigned to each asset (W). Grounded in these ideas, the process of selecting the optimum portfolio can be summarized in the following steps:

1. Specification of a body of assets and investment funds to be considered for the portfolio screening.
2. Asset analysis by means of the estimation of their expected payoffs, variances and covariances.
3. Determination of the investor's efficient frontier and indifference curves.
4. Derivation of the optimum portfolio.
5. Analysis of the Risk Evaluation (VaR) of the optimum portfolio.

These steps are briefly described in Section 2. In Section 3 a description of the system is made. Section 4 deals with the neural network training, Section 5 with the experiments and finally the conclusions of this work are shown in Section 6.

2 Portfolio Selection

Let there be an investor with a budget to be employed in the buying of assets to maximize their expected utility. The Stock Exchange will provide him or her with a lot of investment choices, as many as shares. Nevertheless, the investor must determine the share combination which, while maximizing the proposed objective, uses up the entire available budget. That is, he or she must know what assets to buy and how much to spend on each one of them. To solve this problem we take the following steps:

2.1 Asset Analysis

Following Markowitz, the first step starts with observation and experience and finishes with some specific ideas on the future behaviour of the available assets, particularly concerning the density functions of the future payoffs of the shares.

2.2 Computing the Historical Payoff

Let us see how to compute the historical payoff (R_{it}) of an asset i in a given period of time t. Let $P_{i(t-1)}$ be the price of *asset i* at the end of period *t-1*, that is, at the beginning of period t. Assuming that we buy the share at this moment, it will be the purchase price. Let d_{it} be the cash-flow obtained by the asset in period t. Finally, let P_{it} be considered as the price of the share at the end of period t or, in our case, its selling price. The payoff obtained in period t will be computed as in Eq.1:

$$R_{it} = \frac{P_{it} - P_{i(t-1)} + d_{it}}{P_{i(t-1)}} \tag{1}$$

2.3 Derivation of Efficient Border

Once the individual features of each asset are known, we study features that will comprise the portfolio. For this purpose, we will assume that we have n possible assets, each of them with its mean and variance, as representative of its payoff and risk. A portfolio is a set of assets so it will also have a payoff and variance different from those of its components. Portfolio payoff, R_c will be a function of the different random variables of payoff of the constituent assets and thus will itself be a random variable. Let us compute the risk. To this end, we will compute the portfolio payoff variance $V(R_c)$ as a function of the assets payoff variance σ_i^2, as in Eq.2:

$$V_c = [\Sigma_{i=1}^n R_i] = \Sigma_{i=1}^n w_i^2 \sigma_i^2 + \Sigma_{i=1}^n w_i w_j \sigma_{ij} = \Sigma\Sigma_{i=1} w_i w_j \sigma_{ij} \tag{2}$$

That is, the portfolio payoff variance will depend on the covariances of the assets payoffs.

2.4 Computing the Optimum Portfolio

Once the expected values and variance (risk) of payoff are known, we must decide on the optimum portfolio to choose. We will follow the process defined in the mean-variance rule: compute the efficient portfolios and select the portfolio that maximizes the utility of the decision maker. There are several ways to compute the efficient portfolio borders. Markowitz proposes, among others, the following one, maximize R_c, produced in Eq.3:

$$R_c = \Sigma_{i=1}^n w_i R_i. \tag{3}$$

As is apparent, the problem is approached in terms of quadratic programming where a minimum for an investment risk has to be found, with a given payoff level and no possibility of debt.

2.5 Analysis of Risk Evaluation of Optimum Portfolio

The concept of Risk Evaluation (VaR) [10] comes from the need to measure with some level of confidence the percentage of loss that a portfolio will undergo in a predefined time. This is an estimation of the maximum loss that the portfolio can have. In the system implemented, the VaR is calculated for each asset by the Normal Delta Method [12], chosen because it is considered the simplest one to estimate since it requires only the market values, the final portfolio positions and the variance-covariance matrices. To calculate VaR the steps are:

1. Identify the purchase value for each asset.
2. Check that the changes in the associated values for the asset follow a normal distribution.
3. Compute the variance and covariances for each portfolio asset.
4. Compute the portfolio variance.
5. The VaR is calculated by multiplying the portfolio variance by the corresponding factor to the confidence level (1.65 in this case)

Thus, the VaR is a useful tool in that it gives investors a more precise criterion for judgeing the work done by portfolio managers. Furthermore, it allows us to monitor and control the risks through time by verifying that the short term portfolio does not diverge from the long term objectives.

3 System Description

Based on this theoretical background, a system was developed for the automatic, efficient management of investment fund portfolios which takes into account the history over a given period, adapts to dynamic market conditions and upgrades itself via the web every fortnight. It is assumed that the investor owns an amount of money to be spent and he or she can keep the investment for a certain amount of time. With these premises, the system must suggest to the investor what assets to buy and the amount to be invested in each one to obtain a bigger payoff and a lower risk. Besides that it must indicate what is most suitable for the asset according to the daily evolution of prices and payoffs of each asset: keep, sell or buy. In summary, the aforementioned steps are implemented. These are explained herewith:

3.1 Specification of the Asset Set

The system can work with any portfolio on the stock-exchange market, so there are as many possibilities as assets. To perform the prediction computations, for each asset in the portfolio, a data base is defined with the following fields: ISIN code (fund registration), name of the asset, estimated time in portfolio (inversely dependent on risk), date of portfolio participation, value of participation, payoff, equivalent yearly rate (APR), observed volatility, market distribution and number of days to take into account.

3.2 Analysis of Assets Through Estimation of Expected Payoffs, Variances and Covariances

With the previous data the historical payoff is computed for each asset for a period of 321 days, and the following values are obtained: daily payoff (with respect to previous day), daily volatility (standard deviation), average daily payoff, daily profit or loss and VaR for each asset.

3.3 Determination of the Efficient Border

With the results obtained in the historical payoff phase, minimum variance point (MVP) is determined inside the boundary of production possibilities. To do this, the assets are initially given random weights and two restrictions are implicitly imposed:

1. The client has to spend 100 % of the available money.
2. No negative weights are allowed.

The user has the possibility to add his or her own restrictions. For the entire portfolio the MVP is computed with the Solver Excel tool. This gives us the minimum standard deviation of the portfolio (minimum risk). Later on, the portfolio average daily payoff (MRP) is calculated. This is the Sharpe Ratio simplified by considering that the risk free interest rate is 0 % (this is the case with the Goverment Bonds). Finally the MRP/MVP ratio is computed (maximum slope of the straight line) to maximize the payoff/risk ratio or, equivalently, maximum payoff at a minimum risk.

In summary, we try to find the right weights for each one of the portfolio components so that the agent can choose the best distribution. Once the payoff and risk are calculated, we select the efficient portfolio and compute the VaR.

4 Neural Network Classification

In [2] it is pointed out that the market knows everything. In consequence, we must always study it as a source of maximum available information and thus take decisions of buying or selling the stock. It is not necessary to know all this information: we must simply study the evolutions of prices that are formed. The evolutions will indicate to some degree the likely direction that the prices are going to take in the future, since the market remembers the formations of prices that have taken place throughout history and, they will probably occur again with identical consequences on most occasions.

As soon as all the combinations of the list of assets are obtained, in order to guarantee maximum profitability and the minimal risk, it would be desirable to be able to classify the state of the price in a certain period, bearing in mind its behaviour in a previous period and to be able to know if it goes down, up or keeps constant within the fixed period. It might help the investor to take a decision to buy, sell or do nothing.

With this aim, for every asset we train a perceptron neural network with a single hidden layer [6]. For our case the significant input parameters are the value of daily participation, payoff and daily payoff. With them the net is trained to learn the behaviour of the prices in a one-year period, classifying them into three classes according to their daily profitability: Class 0 (do nothing), Class 1 (sell), Class 2 (buy).

In the training phase we use 70 % of the available information and the remaining 30 % is used for the validation. The net has three input neurons, corresponding to the significant input attributes and three output neurons (classes). The number of neurons in the hidden layer is a parameter to play with to achieve a tradeoff between efficiency and speed of training. In our case, with three neurons an acceptable result is reached.

Once the architecture of the net is defined, we train it using the Weka tool [13]. To do this it is necessary to fix some parameters that takes part in the training process. These parameters always depend on the problem to be solved and after performing some simulations the learning rate is fixed at 0.3 the momentum at 0.1 and the number of training cycles is 30. After training, we perform an estimation of the results provided by the network through the test patterns, and we verify that the number of examples correctly classified depends on the fund in question, ranging between 96 % and 100 %, as we show in the results of the experiments, the error in the estimation of the classes being 0.03.

We observed that the network correctly classified the validation pattern. Once the net has been trained with the prices and final earnings, it can be consulted with any other input value in future periods, and they will be classified to help in the decision making on an asset.

5 Experiments

The data sets for performing the forecasting study of profitability and risk in a portfolio of values uses the 14 funds of different managers of the above mentioned values that were taken from the Fibanc Mediulanum Banking Group Platform All Funds [4]. The number of days to bear in mind is determined by the least amount of all the observations of each one of the 14 founds. For the particular study we used the set of assets appearing in Table 1:

Let us suppose that the investor has decided to invest 57000 Euros. The first thing to do is to randomly distribute this amount among 14 investment assets in order to calculate the profitability and the volatility of this distribution in the portfolio. Later the computations previously mentioned in point 3.3 are performed according to the fixed restrictions. The weights which maximize the portfolio of each of the considered assets are found. To obtain the results we click directly on the graph shown in Figure 1 where there appears a series of random points that calculate automatically:

1. The MVP, which is represented in Figure 1 with a continuous line and it represents the minimum risk.

2. The maximum MRP/MVP ratio (discontinuous line), maximum profitability with minimum risk.

With this information we can obtain the ideal portfolio.

Table 1. Set of assets used in the experiments

Name_Assets	Name_Assets
Franklin H.Y. "A"	Dexia eq l aust "C"
Dws Invest Bric Plus	Ubam Us Equi Value A
Aberdeen Asia Pac "A"	Sch Eur Dyn Grwth A
Fortis l Eq Turk "C"	space Newton Hig Inc
Cre Suis.Cap Jp "H"	Ing(l)inv Eur h.d "X"
Challenge Country Mix (S)	Challenge Financial Fund(S)
Challenge Germany Equity	Fidelity Eur S.C. "E"

5.1 Experimental Results

Taking into account the history of observations in a fixed period for 14 assets and the previous calculations, Table 2 shows the final amount to be allocated to each asset and the time that it must remain in the portfolio to obtain a bigger payoff and lower risk.

Table 2. Final amount and time necessary for efficient portfolio

Name	Amount	Time	Payoff	Risk	%Portfolio
Franklin H.Y. "A"	17100	10	5.67	0.122	30.00
Dws Invest Bric Plus	4	22.03	1.569	8.00	
Aberdeen Asia Pac "A"	5 125	5	26.71	0.960	9.00
Fortis l Eq Turk "C"	0				
Cre Suis.Cap Jp "H"	4 000	5	8.28	1.532	7.00
Challenge Country Mix	3600	10	8.67	0.707	6.30
Challenge Germany Equity	1821	3	20.72	0.987	3.20
Dexia eq l aust "c"	0				
Ubam Us Equi Value A	483	1	19.44	0.925	0.80
Newton Hig Inc	4842	5	27.44	0.997	8.50
Ing(l)inv eur h.d "x"	5 059	5	17.01	0.744	8.87
Challenge Financial Fund	6 800	5	12.54	0.725	11.93
Fidelity "E"	0				

As can be seen, the amount of Euros is very different from the one initially assigned. In Table 2 it appears beside the amount that it is necessary to invest in each asset and the time to keep it, the APR profitability in one year that it is possible to obtain and the volatility in 321 days. The assets whose final

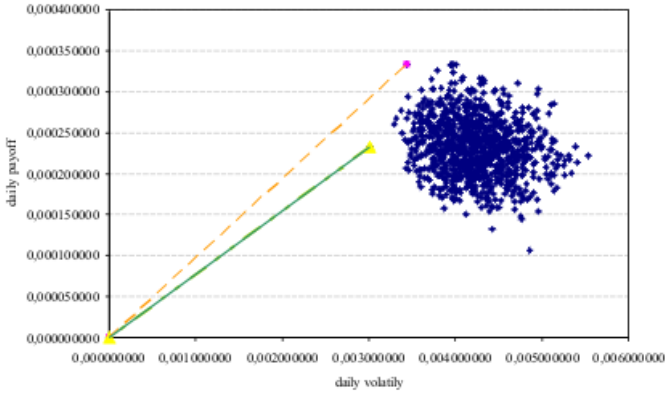


Fig. 1. The graph shows maximum profitability with minimum risk

amount is 0 Euros, are those not recommended to buy. The system also returns the estimated profitability in 321 working days, ensuing from 8 % and from 11 % in one year which means 4836.44 Euros. The VaR analysis is shown in the Table 3.

Table 3. VaR Analysis for portfolio

Profitability estimated in one year	11.00(%)
Daily VaR (in the worst case, the investor will lose)	0.44,(%)
VaR in 321 days	1.72(%)
VaR in one year	3.27(%)
Daily maximum variation assuming 95(%) of the days	251.36

For neural network classification, we performed two fundamental experiments, consistenting of training a neural network for every fund and another one with the information of all the funds in the same period used in the analysis of the portfolio. The worst results on the number of examples classified correctly were obtained by the net that included all the funds for the analyzed period, which could only correctly classify 95.24 % of the cases presented. With one different net for each fund the results range from 96.90 % corresponding to the Fidelity Fund up to 100 % of the majority, as can be seen in Table 4.

The results obtained by means of neural networks were contrasted with those derived from a statistical method. Several approaches were considered based on statistical time series processing and curve adjustments. Results were poor so our conclusion was to use nonparametric approaches, like neural networks, which can learn and adapt to new conditions. The classification errors with Neural Networks were much better in all the cases.

Table 4. Examples correctly classified for each asset

Name_Assets	Precision	Name_Assets	Precision
Franklin H.Y. "A"	100.00	Dexia eq l aust "c"	98.25
Dws Invest Bric Plus	100.00	Ubam Us Equi Value A	100.00
Aberdeen Asia Pac "A"	97.90	Sch Eur Dyn Grwth A	100.00
Fortis l Eq Turk "C"	100.00	Newton Hig Inc	100.00
Cre Suis.Cap Jp "H"	98.30	Ing(l)inv Eur h.d "x"	100.00
Challenge Country Mix (S)	98.60	Challenge Financial Fund	100.00
Challenge Germany Equity	97.30	Fidelity Eur S.C. "E"	96.90

6 Conclusions

A system was created on a formal theorical basis which automates the forecast of the profitability and risk of a portfolio of assets over a one year period, by adopting a strategy that guarantees high profitability and minimal risk for the investor, without restriction in the number and types of assets.

This model offers a methodology for the composition of efficient portfolios, becoming a basic tool for investment decision making. The financial adviser, according to the type of investor (risk adverse, average risk or risk lover), can offer a scale of portfolios with a certain yield, in view of risk level.

The system is able to suggest the asset the investor should buy and the time that it must remain in the portfolio to be profitable. As a consequence, this management is more efficient and achieves better results. Moreover the computer system makes the numerous calculations for the application of the models governing the management mentioned above, as well as the periodic upgrading of the information bases. This system can adapt itself to new trends, since it keeps training with new information, so it can therefore adapt to dynamical market conditions taking into account the good results of previous periods.

The use of neural networks provides an interesting alternative to the statistical classifier. With the results described in previous tables it is clearly shown that with the neural networks classifiers a high level of accuracy can be achieved.

Acknowledgments. This work has been financed by the Spanish Junta de Castilla y León, through the proyect SA064A07.

References

1. Allen, F., Karjalaine, R.: Using Genetic Algorithms to Find Technical Trading Rules. Journal of Financial Economics, 245–271 (1999)
2. Codina, J.: Manual de Análisis Técnico (5ta. Edición). Inversor Ediciones, S. L. Madrid (2007)
3. Dempster, M.: Computational Learning Techniques for Intraday fx Trading Using Popular Technical Indicators. IEEE Transaction on Neural Networks, 744–754 (2001)

4. Fibanc Mediulanum Banking Group, www.fibanc.es
5. Gestel, T., Suykens, J.: Financial Times Series Prediction Using Least Squares Support Vector Machines Within the Evidence Framework. IEEE Transactions on Neural Networks, 809–820 (2001)
6. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice-Hall, Englewood Cliffs (1999)
7. Jorion, P.: Value at Risk: The New Benchmark for Controlling Market Risk. McGraw-Hill, New York (2000)
8. Kodogiannis, V., Lolis, A.: Forecasting Financial Times Series. Using Neural Network and Fuzzy System-Based Techniques Neural Computing & Applications, 90–102 (2002)
9. Markowitz, H.: Portfolio Selection: Journal of Finance 7 (1952)
10. Minnich, M.: A Primer on VaR Perspectives on Interest Rate Risk Management for Money Managers Traders (1998)
11. Tino, P., Schittenkopf, C.: Financial Volatility Trading Using Recurrent Neural Networks. IEEE Transactions on Neural Networks, 865–874 (2001)
12. Vilariño, A. (ed.): Tubulencias Financieras y Riesgos de Mercado. Prentice Hall, Englewood Cliffs (2001)
13. WEKA, http://www.cs.waikato.ac.nz/~ml/weka/index.html

Partitioning-Clustering Techniques Applied to the Electricity Price Time Series

F. Martínez-Álvarez[1], A. Troncoso[1], J.C. Riquelme[2], and J.M. Riquelme[3]

[1]Area of Computer Science. Pablo de Olavide University, Spain
{fmaralv,ali}@upo.es
[2]Department of Computer Science. University of Seville, Spain
riquelme@lsi.us.es
[3]Department of Electrical Engineering. University of Seville, Spain
jsantos@us.es

Abstract. Clustering is used to generate groupings of data from a large dataset, with the intention of representing the behavior of a system as accurately as possible. In this sense, clustering is applied in this work to extract useful information from the electricity price time series. To be precise, two clustering techniques, K-means and Expectation Maximization, have been utilized for the analysis of the prices curve, demonstrating that the application of these techniques is effective so to split the whole year into different groups of days, according to their prices conduct. Later, this information will be used to predict the price in the short time period. The prices exhibited a remarkable resemblance among days embedded in a same season and can be split into two major kind of clusters: working days and festivities.

Keywords: Clustering, electricity price forecasting, time series, day-ahead energy market.

1 Introduction

Due to the Spanish electricity-market deregulation, a will of obtaining optimized bidding strategies has recently arisen in the electricity-producer companies [13]. In that way, forecasting techniques are acquiring significant importance. Thus, this research lies in extracting useful information of the prices time series by using clustering techniques. In this work two well-known clustering techniques [15], K-means and Expectation Maximization (EM), are applied to prices time series in order to find those days which show a similar behavior. These labeled days will be used to forecast the day-ahead price in future work.

Several forecasting techniques have already been used in forecasting miscellaneous electricity time series recently. Indeed, A. J. Conejo et al. [2] used the wavelet transform and ARIMA models and R. C. García et al. [4] presented a forecasting technique based on a GARCH model for this purpose. A mixing of Artificial Neural Networks and fuzzy logic were proposed in [1], while an adaptive non-parametric regression approach is handled in [17]. A model based on

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 990–999, 2007.

the Weighted Nearest Neighbors methodology is presented in [14]. With the aim of dealing with the spike prices, [6] proposed a data mining approach based on both support-vector machine and probability classifier. In [5] mixed models were proposed to obtain the appropriate length of time to use for forecasting prices.

However, none of them used clustering techniques applied to prices time series as a previous stage. The novel and main contribution of this paper is to apply clustering to the electricity prices time series in order to discover behavior's patterns, as a first step to improve forecasting techniques. Therefore, this work tackle the problem in a framework based on non-supervised learning techniques, which will enhance the prices prediction accuracy. The input data is the hourly variation of the price of the electricity throughout the day and is available on [12].

The rest of the paper is organized as follows. In Section 2 the algorithms used, K-means and EM, are described. It is also discussed the number of clusters selected for the analysis. Section 3 shows the results obtained by each method, giving a measure of the quality of them. Finally, Section 4 expounds the conclusions achieved and gives the clues for future work.

2 Partitioning-Clustering Techniques

It has been already demonstrated that partitioning-clustering techniques perform better classifications than fuzzy clustering when electricity prices are considered [11]. In this section two methods are presented, K-means and EM, in order to choose the best algorithm among the partitioning ones. The number of clusters to be generated is one of the most critical parameters, insofar as a too high number could turn the results unclear and muddle the pattern recognition up. Consequently, this optimal number will be widely discussed for each algorithm.

2.1 K-Means Clustering Technique

K-means [10] is a fast method to perform clustering. The basic intuition behind K-means is the continuous reassignment of objects into different clusters so that the within-cluster distance is minimized. It uses an iterative algorithm divided in two phases to minimize the sum of point-to-centroid distances, over all K clusters. The procedure can be summarized as follows:

1. *Phase 1.* In each iteration (evaluation of all the points) every point is reassigned to their closest cluster center. Then the clusters centers are recalculated.
2. *Phase 2.* Points are reassigned only if the sum of distances is reduced. The clusters centers are recalculated after each reassignment.

Selecting the number of clusters. The *silhouette function* [7] provides a measure of the quality of the clusters' separation obtained by using the K-means algorithm. In an object i belonging to cluster C_k, the average dissimilarity of i to all other objects of C_k is denoted by $c_k(i)$. Analogously, in cluster C_m,

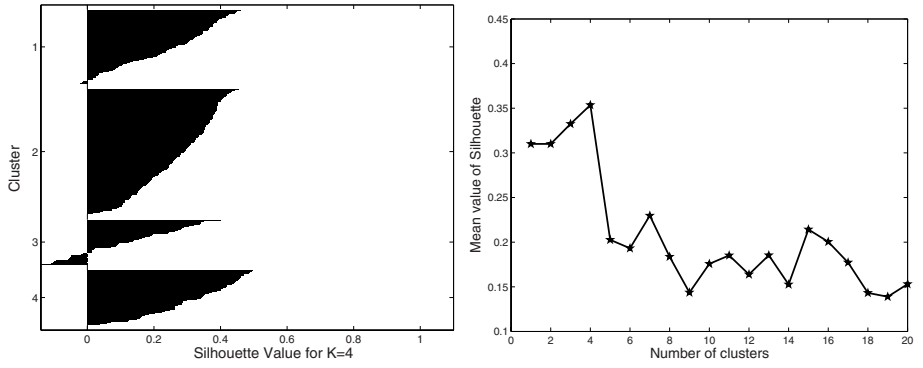


Fig. 1. Silhouette function when $K = 4$. Clusters 2 and 4 are perfectly separated as no negative values were obtained, while clusters 1 and 3 present some uncertainty. The right picture shows the mean value of *silhouette* when varying K.

the average dissimilarity of i to all objects of C_m is called $dis(i, C_m)$. After computing $dis(i, C_m)$ for all clusters $C_m \neq C_k$, the smallest one is selected: $c_m(i) = min\{dis(i, C_m)\}$, $C_m \neq C_k$. This value represents the dissimilarity of i to its neighbor cluster. Thus, the silhouette $silh(i)$ is given by the following equation:

$$silh(i) = \frac{c_k(i) - c_m(i)}{max\{c_k(i), c_m(i)\}} \tag{1}$$

The $silh(i)$ can vary between -1 and $+1$, where $+1$ denotes clear cluster separation and -1 marks points with questionable cluster assignment. If cluster C_k is a singleton, then $silh(i)$ is not defined and the most neutral choice is to set $silh(i) = 0$. The objective function is the average of $silh(i)$ over the N objects to be classified, and the best clustering is reached when the above mentioned function is maximized.

The metric used to determine the *silhouette function,* shown in Figure 1, was the squared Euclidean distance since cosine metrics gave worse results. The maximum mean silhouette value obtained was 0.35, when evaluating the number of clusters from 1 to 20, and it was reached when four clusters were taken into consideration. For this reason [7], the number of clusters selected for further analysis is four ($K = 4$).

2.2 Expectation Maximization

The EM algorithm, proposed by Lauritzen in 1995 [9], is a variation the K-means. The main novelty of this technique is to obtain the previously unknown *Probability Distribution Function* (PDF) [16] of the complete dataset.

This PDF can be approximated as a linear combination of NC components, defined from certain parameters $\Theta = \cup\Theta_j$, $\forall_j = 1...NC$ that have to be found.

$$P(x) = \sum_{j=1}^{NC} \pi_j p(x; \Theta_j) \tag{2}$$

$$\sum_{j=1}^{NC} \pi_j = 1 \tag{3}$$

where π_j are the *a priori* probability of each cluster, $P(x)$ denotes the arbitrary PDF and $p(x; \Theta_j)$ the PDF of each j component. Each cluster corresponds to their respective data samples, which belong to a every single density that are combined. PDF of arbitrary shapes can be estimated by using t-Student, Bernouilli, Poisson, normal or log-normal functions. In this research, the normal distribution has been used as shape of the PDF.

The adjustment of the parameters of the model requires some fitting measure, that is to say, how well fit the data into the distribution. This measure is called data *likelihood*. Therefore, the Θ parameters have to be estimated by maximizing the *likelihood* (ML-Maximum Likelihood criterion) [3]. But what it is usually used is the logarithm of the *likelihood* (*log-likelihood*) because of its easiness to be analytically calculated. The formula of the *log-likelihood* is:

$$L(\Theta, \pi) = log \Pi_{n=1}^{NI} P(x_n) \tag{4}$$

where NI is the number of instances, which are considered to be independent one to another. The EM algorithm, thus, can be summarized in two steps:

1. *Expectation.* It uses the initial values or the ones provided by the previous iteration of the Maximization step in order to obtain different shapes (K-means only finds hyper-spherical clusters) of the desired PDF.
2. *Maximization.* It obtains new parameters values from the data provided in the previous step, maximizing the likelihood measure by using the ML method.

After few iterations, the EM algorithm tends to a local maximum of the L function. Finally, a set of clusters, defined by the parameters of the normal distribution, will be obtained.

Selecting the number of clusters. In the EM algorithm the optimum number of clusters has been obtained with *cross-validation* [8]. The cross-validation method consists in dividing the sample dataset into subsets. The analysis is performed on only one subset while the rest of subsets are used in subsequent confirmation and validation of the initial analysis.

In this research, *V-fold cross-validation* has been used and the original dataset is partitioned into ten subsets or folds ($V = 10$). Only one of these ten subsets is retained as validation data for checking the model, while the remaining nine subsets are utilized as training data. The *cross-validation* process is performed ten times, that is to say, each of the ten subsets are used once as validation data. Finally, the ten results obtained from the folds are averaged and combined to produce a unified estimation. Figure 2 shows the evolution of the logarithm of the likelihood function (log-ML). Thus, the number of clusters selected is eleven since its log-ML value is maximum.

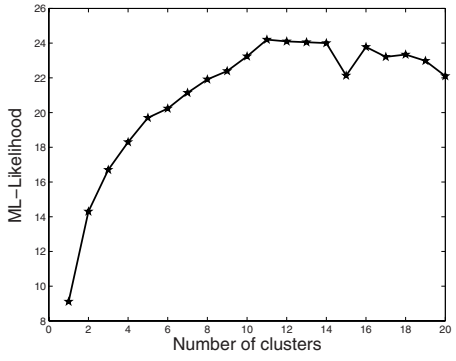


Fig. 2. Justification for the election of the number of clusters with EM

3 Results

The K-means and EM algorithms described in the previous section have been applied in several experiments in order to obtain the forecast of the Spanish electricity price time series for the year 2005 [12].

3.1 K-Means Results

Figure 3 (the left one) shows the year 2005 classified into the 4 clusters. In the x axis are listed the days of the year and in the y axis the cluster to which they belong.

From this automatic classification, two kinds of clusters are easily differentiated. Working days belong to clusters 1 and 2 since they do not contain any Saturday or Sunday. Therefore, the weekends and festivities belong to clusters 3 and 4. This differentiation has been done on the basis of the following criterium. Focusing on samples 10 to 16, it can be appreciated that the 5 first samples (Monday to Friday) belong to cluster 2. On the contrary, samples 15 and 16, Saturday and Sunday respectively, belong to cluster 3 (festivities behave like weekends). This pattern is repeated all the year long but for some samples, whose membership has to be analyzed in detail.

The percentage of membership to the clusters is shown in Table 1.

Although some days seem not to belong to the right cluster, a thorough analysis explains this phenomenon. For example, the 6^{th} day of the year was a Thursday and, according to the previous classification, it should belong to clusters 1 or 2. However, 6^{th} January is a festivity (Epiphany), therefore it behaves as if it was a weekend. For this reason it belongs to cluster 3. This situation is repeated 22 times, that is to say, there are twenty two *working days* that have been grouped in clusters 3 or 4, the clusters associated to weekends and festivities. These days are listed in Table 2.

With regard to weekends, there are six Saturdays that have been grouped as if they were working days, that is to say, they have been classified in cluster 1

Table 1. Distribution of the days in the four clusters created with K-means

Day	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Monday	36.54%	51.92%	3.85%	7.69%
Tuesday	31.48%	57.41%	3.70%	7.41%
Wednesday	30.77%	63.46%	3.85%	1.92%
Thursday	32.69%	59.62%	5.77%	1.92%
Friday	28.85%	59.62%	3.85%	7.69%
Saturday	11.32%	0.00%	39.62%	49.06%
Sunday	0.00%	0.00%	44.23%	55.77%

Table 2. Working days misclassified with K-means

N° of day	Date	Festivity
6	06-01	Epiphany
70	11-03	None
75	16-03	None
77	18-03	Friday pre-Easter
82	23-03	Easter
83	24-03	Easter
84	25-03	Easter
87	28-03	Monday post-Easter
98	08-04	None
122	02-05	Working Day
123	03-05	Madrid Festivity
125	05-05	Long weekend 1^{st} May
126	06-05	Long weekend 1^{st} May
227	15-08	Assumption of Mary
231	19-08	None
235	23-08	None
285	12-10	Columbus Day
304	31-10	1^{st} November long weekend
305	01-11	All Saints'
340	06-12	Spanish Constitution Day
342	08-12	Immaculate Conception
360	26-12	Monday after Christmas

Table 3. Weekends misclassified with K-means

Number of day	Date
169	18^{th} June
176	25^{th} June
183	2^{nd} July
197	16^{th} July
204	23^{rd} July
211	30^{th} July

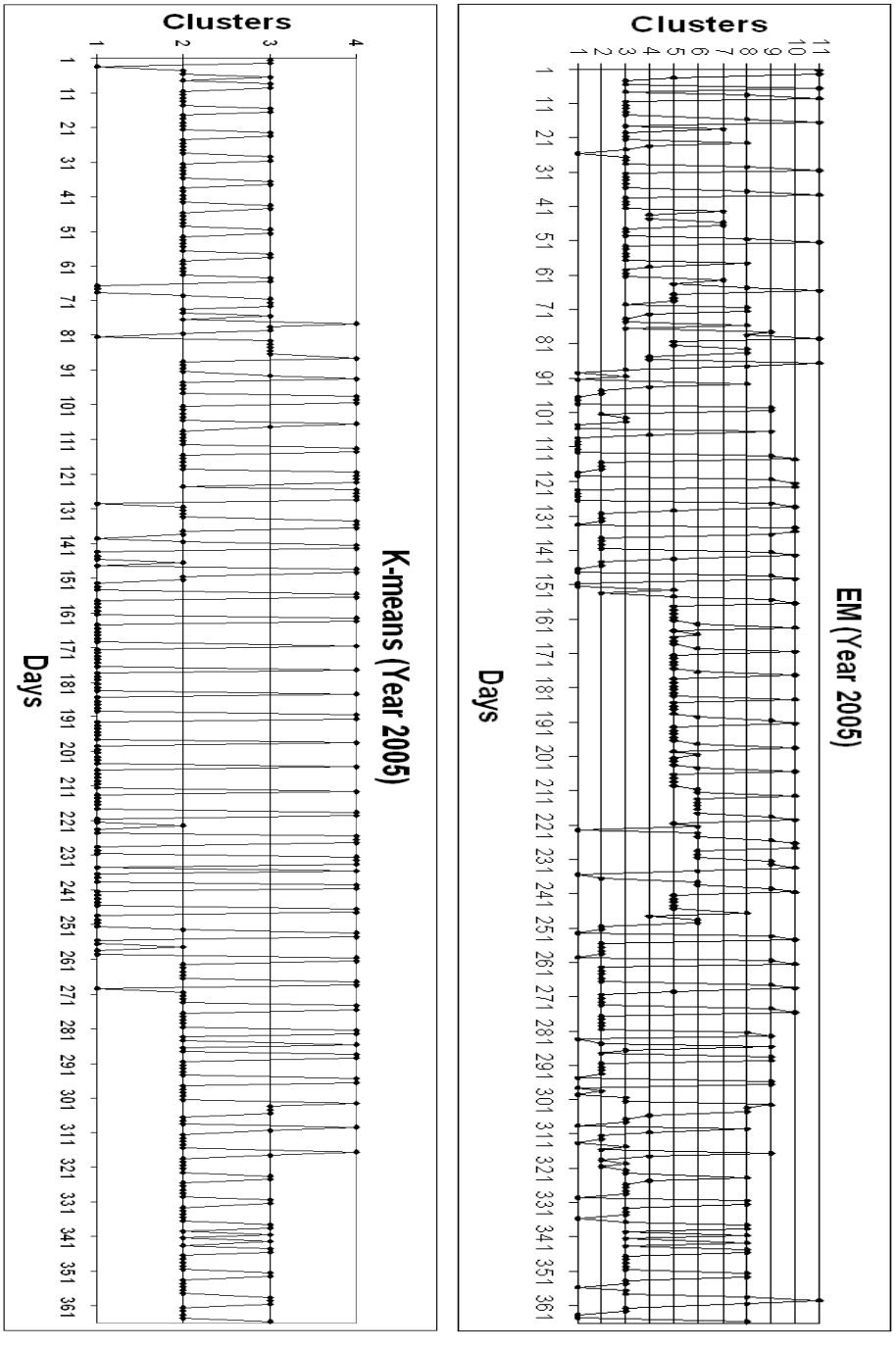


Fig. 3. Distribution of the days belonging to 2005 into the different clusters. The left figure represents the assignation with K-means and the right one with EM.

(one of the clusters identified to belong to the working days) instead of being in either cluster 3 or 4, as it should belong according to the previous classification. These days are listed in Table 3.

The whole year is divided into 261 working days and 104 weekends or festivities. Only five days were not correctly classified (11^{th} March, 16^{th} March, $8t^{th}$ April, 19^{th} August and 23^{rd} August), hence, the average error in working days is 1.92% (5 days out of 261). On the other hand, there were 6 Saturdays improperly grouped. Consequently, the average error for weekends and festivities is 5.77% (6 days of out 104). Thus, the total error is 3.01% (11 days out of 365).

In Figure 3 (the left one) there are three zones clearly differentiated for both working days and festivities. From the 1^{st} January until the 18^{th} May (day number 144), most of the working days belong to cluster 2. From this day until the 20^{th} September (day number 263) they belong to cluster 1. Finally, from the 21^{st} September (day number 264) until the year ends the working days belong again to cluster 2. In festivities there is a similar situation. From the 1st January until the 27^{th} March (day number 86) most of the festivities and weekends belong to cluster 3. From this weekend until 30^{th} October (day number 303) they belong to cluster 4. Finally, from this weekend until the year ends the festivities and weekend belong to cluster 3. Consequently, a seasonal behavior can be observed in the energy prices time series.

3.2 EM Results

Figure 3 (the right one) shows the year 2005 classified into eleven clusters via the EM algorithm. In the x axis are enumerated the days of the year and in the y axis the cluster to which they belong.

From Table 4, it can be stated that the clusters 1, 2, 3, 5 and 7 group clearly the working days since they do not contain any Saturday or Sunday. The clusters 4, 6, 8, 9, 10 and 11 the weekends and festivities, as they hardly contain Mondays, Tuesdays, Wednesdays, Thursdays or Fridays. Further division can be done is this second group. The clusters 4, 10 and 11 are mainly Sundays, while the clusters

Table 4. Distribution of the days in the eleven clusters created with EM

Cluster	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Cluster 1	7.69%	9.62%	15.38%	15.38%	26.92%	0.00%	0.00%
Cluster 2	17.31%	25.00%	23.08%	17.31%	11.54%	0.00%	0.00%
Cluster 3	25.00%	28.85%	30.77%	34.62%	25.00%	0.00%	0.00%
Cluster 4	0,00%	1.92%	0.00%	0.00%	1.92%	3.77%	19.23%
Cluster 5	30.77%	17.31%	21.15%	17.31%	17.31%	1.89%	0.00%
Cluster 6	5.77%	11.54%	3.85%	7.69%	9.62%	11.32%	0.00%
Cluster 7	1.92%	3.85%	0.00%	1.92%	1.92%	0.00%	0.00%
Cluster 8	5.77%	1.92%	3.85%	3.85%	1.92%	39.62%	9.62%
Cluster 9	1.92%	0.00%	1.92%	0.00%	3.85%	39.62%	7.69%
Cluster 10	3.85%	0.00%	0.00%	0.00%	0.00%	1.89%	44.23%
Cluster 11	0.00%	0.00%	0.00%	1.92%	0.00%	1.89%	19.23%

8 and 9 are mainly Saturdays. However, the association of days to clusters with the EM algorithm is not as easy as it resulted with only four clusters. Thus, the dispersion through the clusters is higher. This fact is manifested by a higher error rate since one Saturday and sixteen working day were improperly classified or, equivalently, a 4.38% error rate was committed.

In contrast to what happened in K-means, these sixteen working days do not correspond to weekends or festivities. On the contrary, this misclassification appears randomly and there are no apparent causes. Nevertheless, the Saturday wrong classified (classified into cluster 5) is, like it happened with K-means (see Table 3), the 2^{nd} July: the starting day of holidays for many Spanish people.

4 Conclusions

Partitioning-clustering techniques have been proven to be useful to find patterns in electricity price curves. The analysis carried out via both K-means and Expectation Maximization algorithms yielded relevant information insofar as they found patterns in price time series' behavior.

The average error committed in their classification was 3.01% (11 days) with K-means and 4.38% (16 days) with EM, which means a great degree of accuracy. K-means has been confirmed to be the algorithm more suitable for daily prices classification. Several factors that affect the prediction by increasing the error rate has been identified, such as the time of the day, the day of the week and the month of the year.

Future work is directed to the prediction of day-ahead prices once known the previous clustering. Therefore, the prices prediction will be handled by means of the information gathered from this clustering and used as a temporal indicator of the time series behavior. The K-means algorithm is used, thus, as a step prior to forecasting. Eventually, a label-based algorithm will be proposed with the aim of taking advantage of this extracted knowledge.

Acknowledgments. The authors want to acknowledge the financial support from the Spanish Ministry of Science and Technology, projects TIN2004-00159 and ENE-2004-03342/CON, and from the Junta de Andalucía, project P05-TIC-00531.

References

1. Amjady, N.: Day-ahead price forecasting of electricity markets by a new fuzzy neural network. IEEE Transactions on Power Systems 21(2), 887–896 (2006)
2. Conejo, A.J., Plazas, M.A., Espńola, R., Molina, B.: Day-ahead electricity price forecasting using the wavelet transform and ARIMA models. IEEE Transactions on Power Systems 20(2), 1035–1042 (2005)
3. Cramér, H.: Mathematical methods of statistics. Princeton Univ. Press (1946)
4. García, R.C., Contreras, J., van Akkeren, M., García, J.B.: A GARCH forecasting model to predict day-ahead electricity prices. IEEE Transactions on Power Systems 20(2), 867–874 (2005)

5. García-Martos, C., Rodríguez, J., Sánchez, M.J.: Mixed models for short-run forecasting of electricity prices: Application for the spanish market. IEEE Transactions on Power Systems 22(2), 544–552 (2007)
6. Guha, S., Rastogi, R., Shim, K.: A framework for electricity price spike analysis with advanced data mining methods. IEEE Transactions on Power Systems 22(1), 376–385 (2007)
7. Kaufman, L., Rousseeuw, P.J.: Finding groups in Data: an Introduction to Cluster Analysis. Wiley, Chichester (1990)
8. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 1137–1143 (1995)
9. Lauritzen, S.L.: The EM algorithm for graphical association models with missing data. Computational Statistics and Data Analysis 19(2), 191–201 (1995)
10. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statististics and Probability, pp. 281–297 (1968)
11. Martínez-Álvarez, F., Troncoso, A., Riquelme, J.C., Riquelme, J.M.: Discovering patterns in electricity prices using clustering techniques. In: Proceedings of the International Conference on Renewable Energies and Power Quality (2007)
12. Spanish Electricity Price Market Operator, http://www.omel.es
13. Plazas, M.A., Conejo, A.J., Prieto, F.J.: Multimarket optimal bidding for a power producer. IEEE Transactions on Power Systems 20(4), 2041–2050 (2005)
14. Troncoso, A., Riquelme, J.C., Riquelme, J.M., Martínez, J.L., Gómez, A.: Electricity market price forecasting based on weighted nearest neighbours techniques. IEEE Transactions on Power Systems 22(3), 1294–1301 (2007)
15. Xu, R., Wunsch II., D.C.: Survey of clustering algorithms. IEEE Transactions on Neural Networks 16(3), 645–678 (2005)
16. Zacks, S.: The theory of statistical inference. Wiley, Chichester (1946)
17. Zareipour, H., Bhattacharya, K., Cañizares, C.A.: Forecasting the hourly Ontario energy price by multivariate adaptive regression splines. IEEE Transactions on Power Systems 20(2), 1035–1042 (2006)

Time-Series Prediction Using Self-Organising Mixture Autoregressive Network

He Ni and Hujun Yin

School of Electrical and Electronic Engineering, University of Manchester
he.ni@manchester.ac.uk, hujun.yin@manchester.ac.uk

Abstract. In the past few years, various variants of the self-organising map (SOM) have been proposed to extend its ability for modelling time-series or temporal sequence. Most of them, however, have little connection to, or are over-simplified, autoregressive (AR) models. In this paper, a new extension termed, self-organising mixture autoregressive (SOMAR) network is proposed to topologically cluster time-series segments into underlying generating AR models. It uses autocorrelation values as the similarity measure between the model and the time-series segments. Such networks can be used for modelling nonstationary time-series. Experiments on predicting artificial time-series (Mackey-Glass) and real-world data (foreign exchange rates) are presented and results show that the proposed SOMAR network is a viable and superior to other SOM-based approaches.

1 Introduction

Exchange rate forecasting has always been a challenging area of research that has received a great deal of attention. Since the break up of the Bretton-Woods system in 1973, the trend analysis in spot foreign exchange rates has been a recurrent theme among statisticians and econometricians during the last two decades.

A fundamental way is to use the economic theory to underline the structural relations between exchange rate and other variables and to use statistical methods to identify the correlations between the past data and future moves. Researchers have devoted a great deal of effort on these techniques in order to beat the random walk model. However, these econometric and time-series techniques cannot even outperforms the simplest random walk [1]. The reason is that most of the econometric models are linear and used under specific or strict assumptions. For instance, autoregressive maving average (ARMA) models assume a linear relationship between the current value of the variables and previous values of the variable and error terms. The mean and variance of variables need to be a constant overtime.

Due to the recent advances in computational intelligence and computer power, nonparametric models have been used extensively in the last few years with various successes. Exchange rate forecasted by Artificial Neural Networks (ANNs) provide strong evidence in term of out-of-sample forecasting achievements. Many

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1000–1009, 2007.

comparison studies show that ANNs significantly outperform linear ARMA model and native random walk model [2,3,4]. The most widely used techniques so far are the multilayer perceptron (MLP), radial basis function (RBF) networks and recurrent networks. As a regressive method, support vector machines (SVM) have been proposed as a good alternative for MLP in time-series forecasting. SVMs are established on the theory of the structural risk minimisation principle.

The main problem in modelling financial time-series is their non-stationarity. That is the mean and variance of the time-series are changing over time, which implies that the variables switch their dynamics in different regions. It is particularly true in exchange rates due to the amount of nonconstant "information flow". Empirical studies [5] show that the distribution of daily returns[1] is approximately symmetric and leptokurtic (i.e., heavy tailed). One possible explanation for the heavy tailed distribution is that samples are independently distributed as a normal distribution whose mean and variance change over time. Many others argued that observed returns come from a mixture of normal distributions [6,7]. It is not convincing for a single model to capture the dynamics of the entire time-series.

It is reasonable to assume that a time-series locally is a homogeneous model; at least it is true for most cases. A potential solution can be found using the "divide-and-conquer" principle, in which the entire model is divided into several smaller ones [4]. The solutions are then combined to make the final solution. The prediction is thus made only by the best fit local model.

Self-Organising Map (SOM) can be used to partition the input data to smaller regions by associating input data with their unique best-matching units. The area in the input space for which the reference vector is called Voronoi tesselation. Voronoi tesselations partition the input space into disjoint sets. Models can be created by locally fitting to the specific Voronoi tesselations. The topological relationship of local models are maintained as the pre-determined lattice. There are various successes on different applications. For example, Dablemont [8] applied SOM based local models with RBF network as regressor to predict the returns of the DAX30 index. Liu and Xu [9] used SOM based local models to perform PCA on the data from multi-modes. Cao [4] proposed a SVM experts system, which is also based on SOM local models, to predict time-series.

For the SOM to be used for modelling time-series, the consective input points have to be grouped into vectors to form temporal context by means of a window of a pre-fixed length. The information in between the vectors is however lost. Recently, increasing interest arises in the SOM for time-series or sequence processing. Typical methods include Temporal Kohonen Map (TKM) [10], the recurrent SOM (RSOM) [11].

Lampinen and Oja proposed a method based on the SOM, where every unit represents an AR model with its reference vector as the model parameters [12]. The experiments conduced have shown the model can learn to distinguish textures from images. The method in fact is a multiple AR model with the parameters of component models forming topological orders.

[1] A simple logarithm difference transform.

Here, we propose a similar multi-regressive model with a different winner selection rule to reflect the characteristics of homogeneous time-series.

The rest of paper is organised as follows. In section 2, we describe the proposed methodology. Section 3 will present the application of the proposed methodology for prediction of exchange rate. Finally, conclusions will be summarised in Section 4.

2 Methodology

The problem of predicting future value of a stochastic process is closely related to the problem of estimating the unknown parameters of a regressive model. We start from identifying parameters of artificial autoregressive(AR) models.

The target process is assumed to be generated by several independent stationary discrete autoregressive processes. It has many fields of applications, especially in econometrics and automatic control. A number of studies recent focus on modeling such non-stationary process. The model is based on the assumption that the underly process consists of several independent stationary AR processes, referred to as local models. The model can be considered as a mixture of these independent local models, or is regarded to be from one of these local models at a time. Such a model can be expressed as,

$$F(x_t|\digamma_{t-1}) = \sum_{i=1}^{K} \beta_{(i,\mathbf{x})} \Phi_i (x_t - \phi_{i0} - \phi_{i1} x_{t-1} - \ldots - \phi_{ip_i} x_{t-m_i}). \quad (1)$$

where Φ_i is i-th local AR model. In the mixture AR (MAR) model, $\beta_{(i,\mathbf{x})}$ are the mixing parameters; and in the latter case, considered in this paper, $\beta_{(i,\mathbf{x})}$ are selection functions, given as,

$$\beta_{(i,\mathbf{x})} = \begin{cases} 1 & \text{if } \mathbf{x} \in \Phi_i \\ 0 & \text{else} \end{cases} \quad (2)$$

where input vector $\mathbf{x}(t) = [x(t), x(t-1), \ldots, x(t-m_i)]^T$, $\digamma_{t-1}$ represents the information up to time $t-1$, $\{\phi_i\}$ are parameters, K is the number of AR processes and m_i is the order of AR process i.

2.1 Lampinen and Oja's Self-organising AR Models

Lampinen and Oja proposed a method called self-organizing AR map (SOAR) based on a self-organizing map of "neural" units for unsupervised segmentation and classification of 1D and 2D signals [12]. The SOAR models index each neural unit by i and each with weight vector w_i which signifying an AR process. The prediction is based on,

The error $e(t) = x(t+1) - \mathbf{x}^T \mathbf{w}$ is further smoothed by an exponential average over the recent estimation errors v_i has been used instead of immediate estimation errors e_i

$$v_i' = \beta e_i(t) + (1 - \beta)v_i. \tag{3}$$

where β is a smoothing factor. The best matching unit is the one with smallest v_i'.

The winner and its neighborhood units update their weights according to

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) + g(r)e(t)\mathbf{x}(t). \tag{4}$$

where $g(r)$ is a linearly decreasing adaptation rate. The model is shown working well in segmenting the image into texture classes, without priori knowledge about the number of classes or the class models. The authors also presented good experimental results on 1D artificial signal and 2D textures.

However the performance of the SOAR model in finding underlying AR processes is hampered by the inaccuracy and volatile error terms (i.e. the winning rule), despite being smoothed. It cannot guarantee a good estimation of the parameters of the underlying process. Fig. 1 shows a divergence of such a model for an AR(2) process. The initial weights were set randomly (upper figure) or to the true parameters (lower figure).

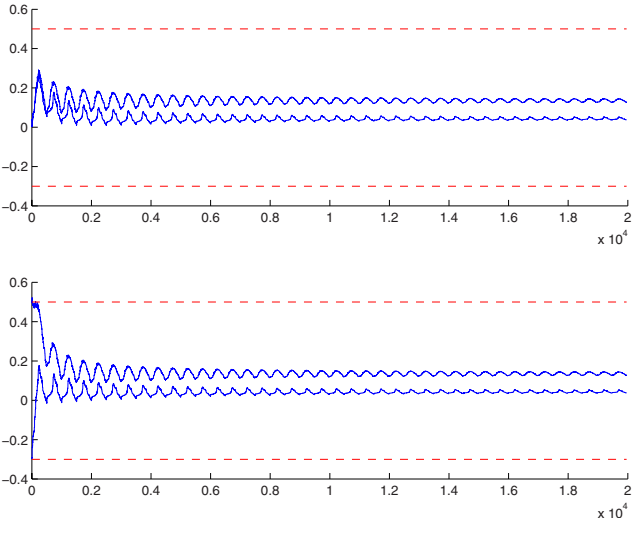


Fig. 1. Estimates of an AR(2) parameters by the SOAR with initial weights set randomly (upper panel), and to the true parameters (lower panel)

2.2 Self-organising Mixture AR Models (SOMAR)

As we assume that the stochastic process is characterised by white noise corruption. As a sufficient condition, the modeling error should be close to the white noise if the modelling is following a "correct" path. Therefore, we investigate the autocorrelation of the error instead of the error itself. In order to obtain sufficient estimation information, we hereby use a small batch input or a patch.

The modelling error is a discrete time-series of length p, the batch size, $\{e(1), e(2), \ldots, e(p)\}$, with mean μ and variance σ^2, an estimate of the autocorrelation coefficient $R(k)$ at lag k can be obtained as

$$R(k) = \frac{1}{(p-k)\sigma^2} \sum_{t=1}^{p-k} (e(t) - \mu)(e(t+k) - \mu). \tag{5}$$

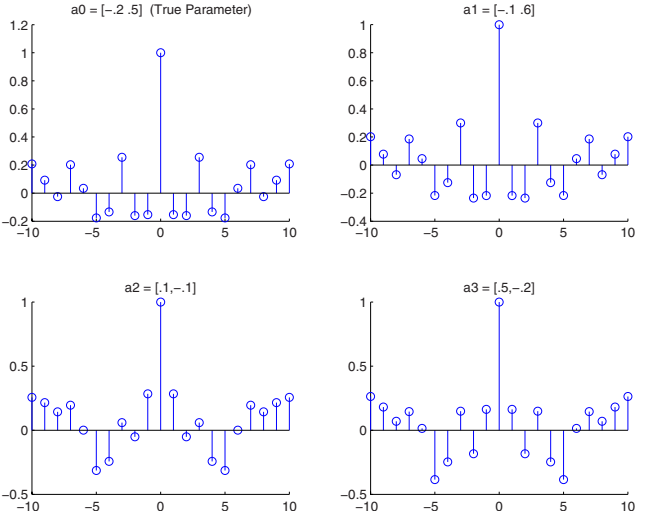


Fig. 2. Autocorrelation of the modelling errors for models $\mathbf{a}_0$ - $\mathbf{a}_3$

Fig. 2 shows the autocorrelations of a set of the modelling errors from a patch of 20 points. The generating parameters are $\mathbf{a}_0$=[-.2, .5]. We randomly test it on three sets of parameters $\mathbf{a}_1$=[-.1, .6], $\mathbf{a}_2$=[.1, -.1] and $\mathbf{a}_3$=[.5, -.2], their correlations are plotted in Fig. 2.

Here we use the sum (of the absolute value) of autocorrelation coefficients (SAC) as the similarity measure,

$$v_i = \sum_{j=-p}^{p} |R_i(j)|. \tag{6}$$

The SAC values for these four cases are 3.8832($\mathbf{a}_0$), 4.3504($\mathbf{a}_1$), 4.5224($\mathbf{a}_2$), and 4.5963($\mathbf{a}_3$) respectively. As we can see that model $\mathbf{a}_1$ is closer to model $\mathbf{a}_0$ that to models $\mathbf{a}_2$ and $\mathbf{a}_3$, so are their SAC values.

In the proposed method, firstly a fixed number of consecutive input vectors are used to make a patch input. Analogy to the SOM algorithm, we choose the winner for that patch input according to the SAC, $v_i, i = 1, 2, \ldots, N$, i is the index of local model and N is the total number of the local models.

Then the winner and its neighbours adapt their weights by

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) + h(r,t)\eta(t)e(t)\mathbf{x}(t). \tag{7}$$

where $h(r,t)$ is the neighborhood function and $\eta(t)$ is a decreasing adaption parameter. Gaussian neighborhood function is used, and linearly decreasing learning rate is used,

$$h(r,t) = e^{-(\frac{r}{2\delta(t)})^2}. \tag{8}$$

$$\eta(t) = \eta_0 \frac{\tau_0}{\tau_0 + \tau_1 t + \tau_2 t^2}. \tag{9}$$

The neighborhood function is a useful feature for SOM alike techniques for avoiding the training process being trapped to local minima, and for forming topology among the nodes.

Here we show a simple example. In total 1,250 consecutive points were constructed by two AR(2) processes $\mathbf{a}_1 = [.5, -.2]$ and $\mathbf{a}_2 = [.4, -.3]$. The signal consists of 5 consequent 250 point long segments. Each segment was generated randomly by one of those two AR processes. The training set is the first 1000 points, shown in Fig. 3 and the testing set is the other 250 points. The sets were divided into 50-point patches.

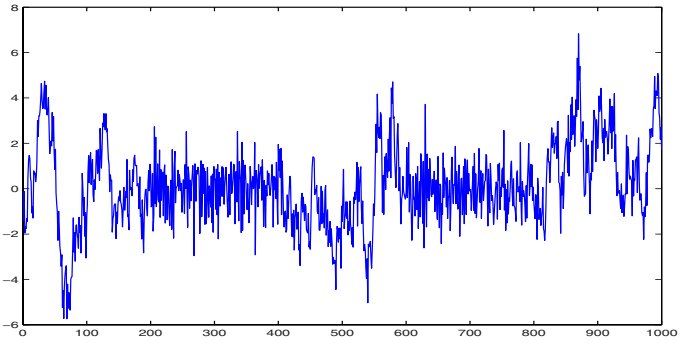


Fig. 3. Training set: 1,000 consecutive points generated by two AR(2) processes $\mathbf{a}_1 = [0.5, -0.2]$ and $\mathbf{a}_2 = [0.4, -0.3]$

The training set was used to train the SOMAR network, the results of the weights are shown in Fig. 4. The trained SOMAR network was tested on the testing set. The results of prediction is illustrated on Fig. 5.

3 Experimental Results

In the section, we present experiments on the artificial data (Mackey-Glass data) and the foreign exchange rate data in respect to the ability of the propose method in characterising the dynamics of non-linear, non-stationary time-series.

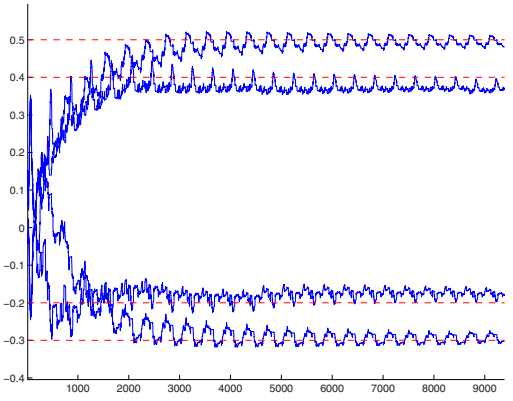


Fig. 4. Parameter estimation of two AR(2) processes $\mathbf{a}_1 = [0.5, -0.2]$ and $\mathbf{a}_2 = [0.4, -0.3]$

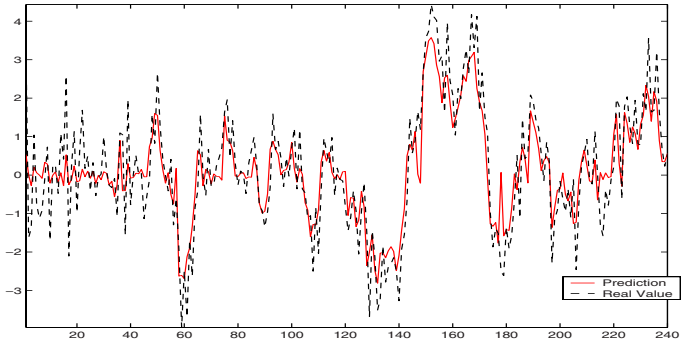


Fig. 5. The dash line represents the original data points, the red solid line represents the prediction by the proposed SOMAR network

3.1 Mackey-Glass Data

To further investigate the capabilities of the proposed SOMAR network, we apply it on a consecutive 600 points Mackey-Glass data, a dynamic system defined by the differential equation:

$$\frac{dx}{dt} = \beta x(t) + \frac{\alpha x(t-\delta)}{1 + x(t-\delta)^{10}}. \tag{10}$$

with the parameter values $\delta = 17, \alpha = 0.2, \beta = -0.1$. We assume the Mackey-Glass data consists of a number of unknown AR processes. In this experiment, the input is the Mackey-Glass series grouped in every 15 points $\mathbf{x}(i) = [x(i), x(i+1), \ldots, x(i+14)]$. We prefixed the order of AR process to 14 in favor of the results of BIC in a previous study [13]. Experiments with other value have been implemented without any significant difference. The result of prediction of Mackey-Glass data shows on Fig. 6.

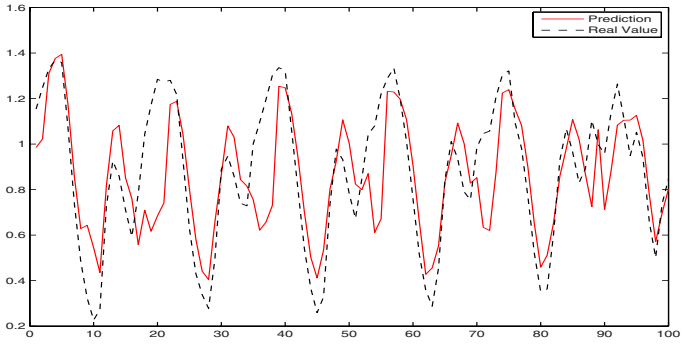


Fig. 6. The dash line represents the original data points, the red solid line represents the prediction by the SOMAR network on Mackey-Glass data

3.2 Foreign Exchange Rate Data

The data was retrieved from the PACIFIC Exchange Rate Service provided by W. Antwiler at UBCs Sauder School of Business. It consists 15 years' daily exchange rates excluding weekends and bank holidays when currency markets were closed. The proposed SOMAR network was trained on 3,000 consecutive data points and the performance of prediction was tested on the following 200 data points. Both the training and testing sets were windowed with the length of 15 points to form input vectors.

For a comparison with other SOM-based methods, we conducted two types of tests as follows.

Predicted FX return. The correct prediction percentage, which is a criterion to check whether the prediction is made on the right direction (i.e. we calculate how many percents predicted returns[2] have the same signs as their corresponding actual returns), shown in Fig. 7.

Predicted FX price. Mean-Square-Error between the testing exchange rates and predicted ones, shown in Fig. 8.

Table 1. Overall predicted FX returns and prices of various methods on the foreign exchange rate data.

	SOMAR	SOAR	VSOM	RSOM
FX return (%)	66.30	54.01	51.00	51.80
FX price	0.0450	0.0601	0.0625	0.0695

The results from two tests are compared to the vector SOM, SOAR and Recurrent SOM, in Table 1. It can be seen that the SOMAR outperforms other

[2] We applied the price-return convert(i.e. $x'_t = \ln \frac{x_{t+1}}{x_t}$ here the x_t is the scalar values of the original data at the time t) to the original data.

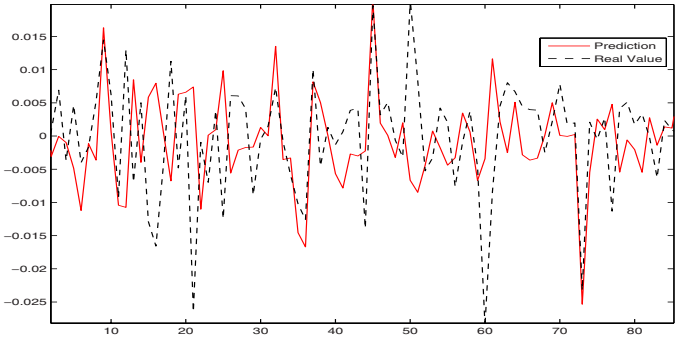


Fig. 7. Predicted returns spanned over 85 days. The dash line represents the FX returns, the solid line represents the prediction by the SOMAR network.

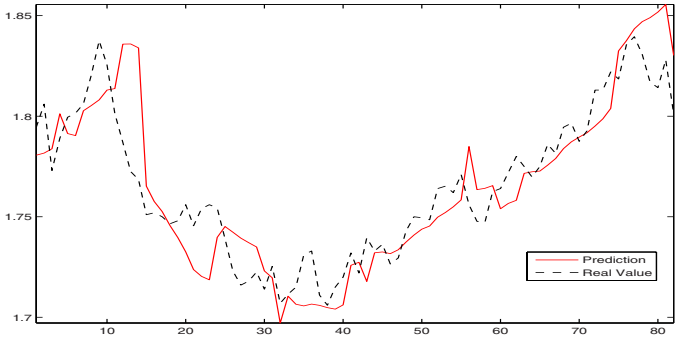


Fig. 8. Predicted rates spanned over 85 days. The dash line represents the FX prices, the solid line represents the prediction by the SOMAR network.

temporal SOM models. The experiments show that SOMAR is a good alternative method to cope with the nonstationarity and multiple underlying processes time-series.

4 Conclusions

A new approach to tackling nonstationarity of real-world time-series has been proposed by using the self-organising mixture autoregressive (SOMAR) model. The model consists of local autoregressive (AR) models and is organised and learnt by a self-organising map, so forming topologically ordered local regressive models. The proposed autocorrelation-based similarity measure makes the network effective and more robust compared to the error-based or Euclidean-based measures. The experiments show that the proposed model can correctly detect and uncover underlying AR models. They also show that the proposed

method outperforms other SOM-based methods in modelling and prediction of nonstationary foreign exchange rates time-series.

References

1. Meese, R.A., Rogoff, K.: Empirical exchange rate models of the seventies: do they fit out of sample? Journal of International Economics 14, 3–24 (1983)
2. Hann, T.H., Steurer, E.: Much ado about nothing? Exchange rate forecasting: Neural networks vs. linear models using monthly and weekly data. Neurocomputing 10, 323–339 (1996)
3. Chen, A.-S., Leung, M.T.: Regression neural network for error correction in foreign exchange forecasting and trading. Computers and Operations Research 31, 1049–1068 (2004)
4. Cao, L.J.: Support Vector Machines Experts for Time Series Forecasting. Neurocomputing 51, 321–339 (2002)
5. Diebold, F.X.: Empirical Modeling of Exchange Rate Dynamics. Springer, New York (1988)
6. Clark, P.K.: A Subordinate Stochastic Process Model With Finite Variance for Speculative Price. Economietrica, 135–155 (1973)
7. Mandelbrot, B., Taylor, H.: On the Distribution of Stock Price Differences. Operations Research 15, 1057–1062 (1969)
8. Dablemont, S., Simon, G., Lendasse, A., Ruttiens, A., Blayo, F., Verleysen, M.: Time series forecasting with SOM and local non-linear models - Application to the DAX30 index prediction. In: WSOM 2003. proceedings - Workshop on Self-Organizing Maps, Hibikino, Japan, pp. 340–345 (September 11-14, 2003)
9. Liu, Z., Xu, L.: Topological local principal component analysis. Neurocomputing 55, 739–745 (2003)
10. Chappell, G., Taylor, J.: The temproal Kohonen map. Neural Networks 6, 441–445 (1993)
11. Koskela, T.: Time Series Prediction Using Recurrent SOM with Local Linear Models. Helsinki University of Technology (2001)
12. Lampinen, J., Oja, E.: Self-Organizing Maps for Spatial and Temporal AR Models. In: Proc. 6th SCIA, Scandinavian Conference on Image Analysis, Helsinki, Finland, pp. 120–127 (1989)
13. Ni, H., Yin, H.: Self-Organising Maps and Local MLP Models for Exchange Rate Prediction. In: Proceedings of the 12th Chinese Automation Computing Society Conference, UK, Loughborough, England (2006)

Adjusting the Generalized Pareto Distribution with Evolution Strategies – An application to a Spanish Motor Liability Insurance Database*

María J. Pérez-Fructuoso[1], Almudena García[3], Antonio Berlanga[2], and José M. Molina[2]

[1] Departamento de Economía de la Empresa
[2] Departamento de Informática
Universidad Carlos III de Madrid, Campus de Colmenarejo
Avenida de la Universidad Carlos III, 22. Colmenarejo 28270. Spain
[3] Departamento de Economía de la Empresa
Universidad de Alcalá de Henares
Plaza de la Victoria, 2. Alcalá de Henares 28802. Spain

Abstract. Management of extreme events is required of a special consideration, as well as a sufficiently wide time horizon for solvency evaluation. Whereas their classical adjustment is usually carried out with Extreme Value Theory (EVT)-based distributions (namely, the Generalized Pareto Distribution), Evolutionary Techniques have been tried herein to fit the GPD parameters as an optimisation problem. The comparison between classical and evolutionary techniques highlights the accuracy of the evolutionary process. Data adjusted in this paper come from a Spanish motor liability insurance portfolio.

1 Introduction

Extreme events are low-frequency, high-severity occurrences that classical risk theory cannot entirely capture, as they take place at the tail of the loss probability distribution [1]. Extremes may give rise to higher fluctuations than classical volatility and uncertainty risks, turning notably complex, thus, to specify both loss amounts and capital sums devoted to their coverage.

An accurate estimation of extreme claims is fundamental to assess solvency capital requirements (SCR). Extreme Value Theory (EVT) (classical parametric estimation) is generally used to fit a Generalized Pareto Distribution (GPD) to excesses over a certain threshold depending on the available data. In this work, data are related with the motor liability insurance historical datasets of one representative company operating within the Spanish market. In this work, an Evolution Strategy (ES) is used to obtain those parameters that better fit the GPD to the experimental data. ES lies in the general field of natural metaphor algorithms (simulated annealing, genetic programming and so on), and could be named as Darwinian approach or evolutionary

* Funded by CICYT TSI2005-07344, MADRINET S-0505/TIC/0255 and AUTOPIA IMSERSO.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1010–1019, 2007.

programming after [2]. In order to allow an evaluation of the characteristics of the new method, some comparisons have been carried out with classical estimation. This method has been applied before to optimize the parameter values of different possible models in order to adjust them to a real data distribution using simple ES [3] and multiobjetive ES [4]. In this case, models define different distributions representing the behavior of reported claims after a catastrophe occurs. The optimization was carried out using real data recorded from several Spanish catastrophes[1].

This new method provides insurers a useful tool to manage risks, allowing them to infer on a statistical basis the extreme values of either a population or a stochastic process, and hence estimate the probability of yet more extreme events than the historical ones. By modeling extremes aside the global sample data, EVT enables insurers to predict both high values at the tail (outliers) and situations exceeding the historical records, without need to turn to the global distribution of the data observed altogether. Consequently, the study of extreme risk preserves insurers' stability and solvency when faced to the occurrence of extreme losses, by application of statistical models to more precisely measure risk and optimally decide on capital requirements, level of provisions, pricing and cession to reinsurance.

In section 2, the definition of GPD and the classical estimation methods are presented. A brief description of ES [5] is included in section 3. Section 4 is devoted to the results of the classical procedures. ES results appear in section 5 and, finally, a comparison with classical techniques and some conclusions are summarized in section 6.

2 Estimation of Generalized Pareto Distribution

The Pickands-Balkema-De Haan theorem demonstrates that a GPD can be adjusted over a certain threshold. For further information on EVT, among others, we refer the reader to [6], [7], [8], [9] and [10].

The distribution of excesses over *u* is defined as:

$$F^u(x) = F^u(y+u) = P\left(X-u \le y / X > u\right) = \frac{F(x)-F(u)}{1-F(u)} \quad for \quad 0 \le y \le x_0 - u \tag{1}$$

where x is the total claim amount, x_0 represents the finite or infinite supreme value of the distribution function, and y stands for the excess over the threshold u, with $y = x - u$.

From (1), once the value of the threshold has been optimized, it is possible to fit $F^u(x)$ to a GPD. The GPD is a two parameter distribution with distribution function:

$$W_{\gamma,\beta}(x) = \begin{cases} 1-\left(1+\xi\frac{y}{\beta}\right)^{-1/\xi} & \text{si } \xi \neq 0 \\ 1-\exp\left(\frac{-y}{\beta}\right) & \text{si } \xi = 0 \end{cases} \tag{2}$$

[1] Data from the reinsurance department of "Consorcio de Compensación de Seguros".

where $x \geq 0$ if $\xi \geq 0$, and $0 \leq x \leq -\frac{\beta}{\xi}$ if $\xi < 0$, with ξ and β being the shape and scale parameters. When $\xi > 0$, we have the usual Pareto distribution with shape $\alpha = 1/\xi$, and the GPD is heavy-tailed. If $\xi < 0$, we have a type II Pareto distribution, whereas $\xi = 0$ gives the exponential distribution.

The family of GPD's may be extended by adding a location parameter, μ. Then the GPD $G_{\xi,\mu,\beta}(x)$ is defined as $G_{\xi,\beta}(x-\mu)$.

The steps to perform the classical theoretical adjustment of the GPD to a extreme data series are the following:

1) Choose the optimum threshold over which the GPD may be fitted to the corresponding values over the threshold, using to that aim the empirical mean excess function.
2) Estimate the parameters of the model according to the heavy-tailedness of the distribution, by means of those estimators that minimize the Mean Squared Error (MSE):

- MLE (GP): Maximum likelihood estimator for parameters ξ and β based on data sets governed by a GPD.
- Moment (GP): Moment estimate for parameter ξ of GPD's (a better name would be Log-Moment estimate). The estimate is related to the Hill estimate and computed with the *k* largest values of the active data set.
- L-Moment (GP): The L-Moment estimator in the full GP model. Take care that the true shape parameter ξ is smaller than 1.
- Drees-Pickands (GP): Drees-Pickands estimate of the parameter ξ of GPD. The estimate is a mixture of Pickands estimates based on the *k* largest values of the underlying sample.
- Slope(GP): The slope β of the least squares line, fitted to the mean excess function right of the *k* largest observations, is close to $\xi/(1-\xi)$, and, therefore, $\xi(k) = \mu/(1+\mu)$ is a plausible estimate of ξ.

3) Check the goodness-of-fit of the underlying distribution through the Quantile-Quantile plot (QQ plot).
4) Perform the inference on the basis of our estimated conditional model, and calculate the marginal probabilities leading to determine the unconditional data distribution.

Our analysis will focus on one representative Spanish insurer' motor liability portfolio along a ten year-period. It exhibits a recent history (from ten years on), although significantly improved over the last years of the interval. We have introduced a distortion in order to keep the company identity undisclosed.

Two different concepts are assumed as forming the loss amount:

- The cost of settled claims, summing all the net recapture payments already made out.
- The cost of non settled claims, comprising all the net recapture payments already made out, and/or the reserves for estimated future payments still pending.

Data have been updated to 2006 values. Table 1 lists below, on an annual basis, the number of claims of the company, with both their total and average individual costs expressed in nominal currency units. Data in Table 1 indicates that the insurer lacks of a stable behavior as regards the average cost evolution. This may be attributed to three main explanatory factors: diverse coverages integrating the final cost of the claims, different claim settlement periods, and the occurrence of extreme events.

The history of the firm (Table 1) shows a weighted loss frequency as high as 45.13 percent during the last four years of the interval, although decreasing until stability with the growth of policies in portfolio.

Table 1. History of the firm

Year	Total cost	Annual variation (%)	Average cost	Annual variation (%)	Number of claims	Annual variation	Number of policies	Annual variation	claims/policies ratio
1	65.487		31,20		2.099		0		
2	182.745	179%	32,08	2,84%	5.697	171,34%	7.913		72,00%
3	284.724	56%	30,29	-5,58%	9.400	65,01%	14.207	79,54%	66,16%
4	375.414	32%	32,45	7,13%	11.569	23,08%	18.454	29,90%	62,69%
5	444.348	18%	32,39	-0,17%	13.717	18,56%	22.300	20,84%	61,51%
6	585.843	32%	35,73	10,29%	16.397	19,54%	28.564	28,09%	57,40%
7	831.326	42%	39,73	11,21%	20.923	27,60%	41.027	43,63%	51,00%
8	1.177.518	42%	42,00	5,69%	28.039	34,01%	59.467	44,95%	47,15%
9	1.408.682	20%	40,93	-2,53%	34.415	22,74%	78.297	31,67%	43,95%
10	1.774.348	26%	44,92	9,74%	39.501	14,78%	93.444	19,35%	42,27%

3 Evolution Strategies

Evolution strategies (ES) developed by Rechenberg and Schwefel, have been traditionally used for optimization problems with real-valued vector representations. As Genetic Algorithms (GA), ES are heuristics search techniques based on the building block hypothesis. Unlike GA, however, the search is basically focused on the gene mutation. This is an adapting mutation based on the likely the individual represents the problem solution. The recombination plays also an important role in the search, mainly in adapting mutation.

Evolutionary Algorithms combine characteristics of both classifications of classical optimization techniques: volume-oriented and path-oriented methods. Volume-oriented methods (Monte-Carlo strategies, clusters algorithms) apply the searching process scanning the feasible region while path-oriented methods (pattern search, gradient descent algorithms) follow a path in the feasible region. A definition of a restricted search space of finite volume and the starting point is required to volume-oriented and path-oriented methods respectively.

Evolutionary Algorithms characteristics change during the evolutionary process and both exploitation and exploration search takes places. ES are techniques widely used (and more appropriated than Genetic Algorithm) in real-values optimization

problems. Evolutionary computation algorithms offer practical advantages facing difficult optimization problems [11]. These advantages are: conceptual simplicity, broad applicability, potentiality to use knowledge and hybridize with other methods, implicit parallelism, robustness to dynamic changes, capability for self-optimization and capability to solve problems that have no known solutions. A general ES is defined as an 8-tuple [12]:

$$\text{ES}= (I, \Phi, \Omega, \Psi, s, \iota, \mu, \lambda) \tag{3}$$

where $I = (\vec{x}, \vec{\sigma}, \vec{\alpha}) = \Re^n \times \Re_+^{n_\sigma} \times [-\pi, \pi]^{n_\alpha}$ is the space of individuals, $n_\sigma \in \{1,\ldots, n\}$ and $n_\alpha \in \{0, (2n-n_\sigma)(n_\sigma-1)/2\}$, $\Phi : I \rightarrow \Re = f$, is the fitness function, $\Omega = \{m_{\{\tau, \tau', \beta\}}: I^\lambda \rightarrow I^\lambda\} \cup \{r_{\{rx, r\sigma, r\alpha\}}: I^\mu \rightarrow I^\lambda\}$ are the genetic operators, mutation a recombination operators. $\Psi(P) = s(P \cup m_{\{\tau, \tau', \beta\}}(r_{\{rx, r\sigma, r\alpha\}}(P)))$ is the process to generate a new set of individuals, s is the selection operator and ι is the termination criterion.

In this work, the definition of the individual has been simplified: the rotation angles n_α have not been taken into account, $n_\alpha=0$.

The mutation operator generates new individuals as follows:

$$\sigma_i' = \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1)) \tag{4}$$

$$\vec{x}' = \vec{x} + \sigma_i' \cdot \vec{N}(\vec{0},1) \tag{5}$$

ES has several formulations, but the most common form is (μ, λ)-ES, where $\lambda > \mu \geq 1$, (μ, λ) means that μ-parents generate λ-offspring through recombination and mutation in each generation. The best μ offspring are selected deterministically from the λ offspring and replace the current parents. Elitism and stochastic selection are not used. ES considers that strategy parameters, which roughly define the size of mutations, are controlled by a "self-adaptive" property of their own. An extension of the selection scheme is the use of elitism; this formulation is called $(\mu+\lambda)$-ES. In each generation, the best μ-offspring of the set μ-parents and λ-offspring replace current parents. Thus, the best solutions are maintained through generation. The computational cost of (μ, λ)-ES and $(\mu+\lambda)$-ES formulation is the same.

4 Problem Resolution with Classical Techniques

The classical GPD adjustment to a data sample involves two fundamental issues:

- The choice of the optimal threshold u, since not all the priorities render a good adjustment of the parametrical function.
- The estimation of the distribution shape (μ) and scale (β) parameters.

The mean excess plot for the 1000 highest claims, with pairs $(X_{k+1}; E_{k,n})$ *for* $k = 1,\ldots,n-1$, is useful to select the threshold (see Figure 1).

The plot is growing linear from a quite low priority through roughly 30000, where the function becomes plain or even decreasing. Then, one may wonder whether the extreme losses are heavy-tailed or not, and therefore, if they may be adjusted to a GPD.

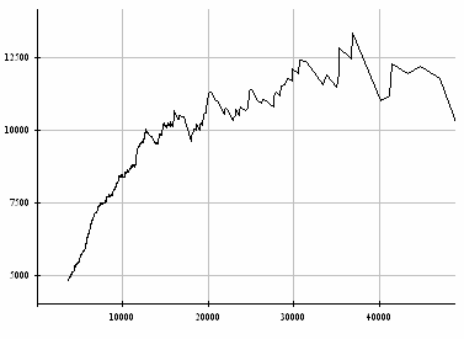


Fig. 1. Mean excess function for the 1000 most severe claims

The QQ plot versus the exponential distribution, by means of the pairs $\left(F_n^{-1}(p); X_{r,n}\right)$, where $F_n^{-1}(p) = -\ln(1-p) = X_p$ and $p = \frac{r}{(n+1)}$, or $p = \frac{r-1/3}{n+1/3}$ provides the answer to both questions. A straight line would mean the goodness-of-fit of the distribution (namely, the exponential distribution), whereas a curve would indicate a more heavy-tailed distribution (Figure 2).

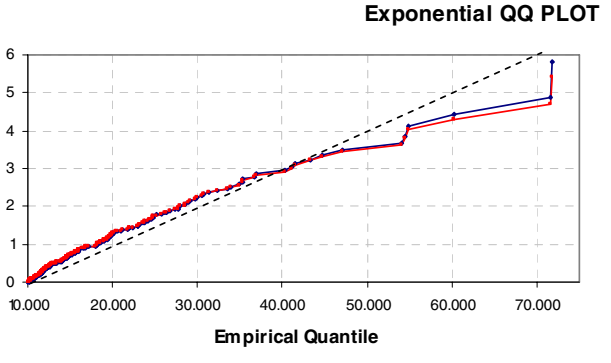


Fig. 2. Exponential QQ Plot

The slight curvature of the empirical quantiles reveals that the tail cannot be modelled with the exponential distribution. But as the plot is very close to linearity, peak losses are heavy-tailed, although not quite. That's to say, the GPD adjusting parameter is positive, but very close to zero.

Being aware that the distribution is heavy-tailed and ignoring at the same time the optimal threshold, we opt by the XTREMES algorithm to both optimize the choice of the threshold to fit the GPD and estimate its parameters. Results are shown in Table 2.

Table 2. Classical Estimation Results

Estimation	Threshold	Tail index ξ	Scale parameter β
MLE	11908	0,137872	8454,29
ME	11908	0,222948	8072,61
L-Moment	11908	0,159826	8220,06
Drees-Pickands	11908	0,154199	8297,47
Slope	11908	0,0433713	8664,96

5 Problem Resolution with Evolution Strategies

The global optimization procedure must adjust the GPD parameters for each value of the threshold minimizing the MSE. The general optimization problem is summarized in the following definition [13]:

Given a function $f: M \subseteq \Re^n \rightarrow \Re$, $M \neq \emptyset$, for $x \in M$ the value $f^* := f(x^*) > -\infty$ is called a global minimum, iff: $\forall x \in M: f(x^*) \leq f(x)$

Then x^* is a global minimum point, f is called objective function, and the set M is called the feasible region. In this case, the global optimization problem has a unique restriction: $\xi > 0$. This restriction is included in the codification and all individuals are processed to become feasible ones. Then, in spite of this restriction, the solutions space does not have infeasible regions.

The type of recombination used in this work is the discrete recombination and the strategy $(\mu+\lambda)$-ES was used to select the individual to the next generation. The parameters of the ES are summarized in Table 3. Besides, different runs where achieved changing the random seed.

Table 3. Setting of exogenous parameters of the ES

Parameter	Value
Initial standard deviations $\sigma_i(0)$	Randomly generated in range [0.01,10.0]
Number of rotation angles n_α	0
Parent population size μ	20
Offspring population size λ	40
Termination criterion	Number of generation step = 500

Results are shown in Figures 3, 4 and 5. In Figure 3, the value of MSE is depicted as function of the threshold value. Each threshold value represents an optimization process to obtain the optimal parameters of GPD for this value. The best value is shown in Figure 3, settled the threshold to 11908. Figure 4 and 5 show the values of parameters for each threshold.

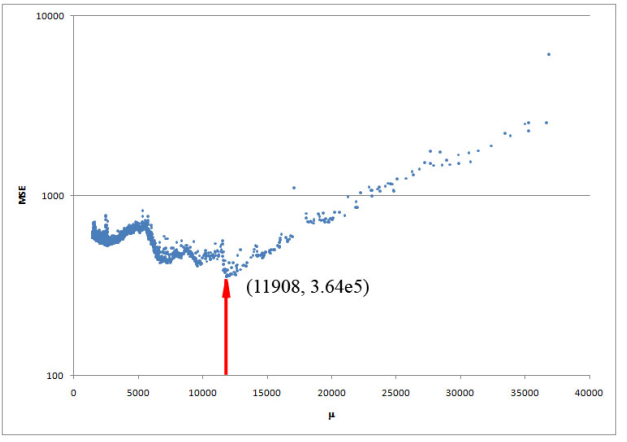


Fig. 3. Optimized MSE value for each threshold

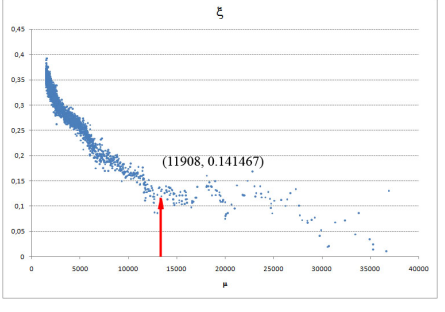


Fig. 4. Optimized ξ for each threshold

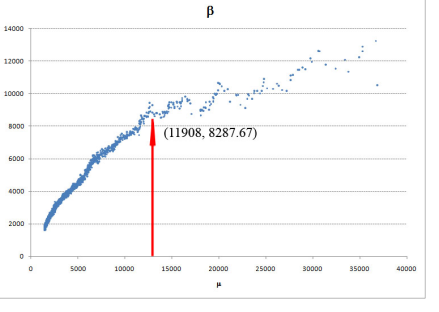


Fig. 5. Optimized β for each threshold

The minimum MSE (3.64E+05) is obtained for a value threshold of 11908. This point is the optimum threshold with values of $\xi = 0.141467$ and $\beta = 8287.67$.

6 Conclusion and Discussion

Reinsurers, as well as cedent insurers, share a deep concern in being able to accurately estimate the probability of claims over a certain threshold. Their expertise in handling extreme risks is decisive to determine how much financial capacity is required to assume the cost of extreme losses, or, instead, cede them to reinsurance. Fitting a GPD to claims above a high threshold is a powerful tool to model the tail of severe losses. We have proposed in this work an ES-based optimization method to estimate the GPD parameters. Table 4 shows a comparative analysis of classical techniques and the proposed method.

Table 4. Setting of exogenous parameters of the ES

Estimation	Threshold μ	Tail index ξ	Scale parameter β	MSE
Evol. Strategies	11908	8287.67	0.141467	3.64E+05
MLE	11908	8454.29	0.137872	3.71E+05
Drees-Pickands	11908	8297.47	0.154199	4.59E+05
L-Moment	11908	8220.06	0.159826	4.88E+05
Slope	11908	8664.96	0.0433713	2.98E+06
Moment	11908	8072.61	0.222948	3.72E+06

Results show that evolution strategies fit the distribution better than EVT classical techniques. We can conclude that ES are an efficient technique to solve the global minimization problem (minimizing the MSE) without any domain knowledge. Then, we probe that ES could be considered in actuarial, financial and economic areas as a solver mechanism able to approximate any mathematical model to a set of real data, as we show in previous works [3],[4] by adjusting the claims rate from a single and multi objective perspective.

References

[1] Këllezi, E., Gilli, M.: Extreme Value Theory for Tail-Related Risk Measures. Kluwer Academic Publishers, Dordrecht (2000)
[2] Holland, J.H: Adaptation in natural an artificial Systems, MIT Press, Bradford Books edition, Michigan, MI (1975)
[3] García, J., Pérez, M.J., Berlanga, A., Molina, J.M.: Adjustment of Claims Rate with Evolution Strategies in a Mathematical Model for Insurance Loss Ratio, Internet, e-com and Artifcial Intelligence. In: III International Workshop on Practical Applications of Agents and Multiagent Systems, Burgos, Spain, pp. 173–182 (2004)
[4] Pérez, M.J., García, J., Martí, L., Molina, J.M.: Multiobjective Optimization Evolutionary Algorithms in Insurance-Linked Derivatives. In: Bernard, J.-F. (ed.) Handbook of Research on Nature Inspired Computing for Economy and Management, vol. II, pp. 885–908. Idea Group Inc, USA (2006)
[5] Schewefel, H.-P.: Evolutionary learning optimum-seeking on parallel computer architectures. In: Sydow, A., Tzafestas, S.G., Vichnevetsky, R. (eds.) Proceedings of the International Symposium on Systems Analysis and Simulation 1988, I: Theory and Foundations, pp. 217–225. Akademie-Verlag, Berlin (1988)
[6] Beirlant, J., Teugels, J.L., Vynckier, P.: Practical Analysis of Extreme Values. Leuven University Press, Leuven (1996)
[7] De Haan, L., Ferreira, A.: Extreme value theory: An introduction. Springer Series in Operations Research and Financial Engineering. Springer, New York (2006)
[8] Embrechts, P., Klüppelberg, C., Mikosch, T.: Modelling extremal events for Insurance and Finance. In: Applications of Mathematics, Springer, Heidelberg (1997)
[9] Kotz, S., Nadarajah, S.: Extreme value distributions. Theory and Applications. Imperial College Press, London (2000)
[10] Reiss, R.D., Thomas, M.: Statistical Analysis of Extreme Values with applications to insurance, finance, hydrology and other fields, 2nd edn. Birkhäuser (2001)

[11] Fogel, D.B.: The Advantages of Evolutionary Computation. In: Lundh, D., Olsson, B., Narayanan, A. (eds.) Proc. of BCEC97: BioComputing and Emergent Computation, pp. 1–11. World Scientific, Singapore (1997)
[12] Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, Inc, Oxford (1996)
[13] Törn, A., Zilinskas, A.: Global Optimization. LNCS, vol. 350. Springer, Heidelberg (1991)

Independent Factor Reinforcement Learning for Portfolio Management

Jian Li, Kun Zhang, and Laiwan Chan

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{jli,kzhang,lwchan}@cse.cuhk.edu.hk

Abstract. In this paper we propose to do portfolio management using reinforcement learning (RL) and independent factor model. Factors in independent factor model are mutually independent and exhibit better predictability. RL is applied to each factor to capture temporal dependence and provide investment suggestion on factor. Optimal weights on factors are found by portfolio optimization method subject to the investment suggestions and general portfolio constraints. Experimental results and analysis are given to show that the proposed method has better performance when compare to two alternative portfolio management systems.

1 Introduction

During the past decade, there have been growing number of researches that apply reinforcement learning (RL) [1, 2] techniques to solve problems in financial engineering [3]. What is of particular interest would be using RL to design financial trading systems. Neuneier used Q-learning algorithm to design a system for trading single asset [4]. The system was enhanced in [5] to enable multi-asset trading. However, the size of action space increases exponentially with the number of assets, hence it requires substantial amount of training data to determine policy for such a huge action space. Ormoneit and Glynn applied a kernel-based RL approach on single-asset trading problem [6]. In [7] Dempster and Leemans proposed a layered single-asset trading system where recurrent RL algorithm is used to offer trading recommendation. The recommendation is then evaluated by risk management overlay to make final decision.

One problem of current researches is that, most works only addressed the simple single-asset allocation problem, i.e. capital can either be kept in cash or invested in a risky asset. In practice, however, investors rarely adopt such an extreme strategy. Instead they normally make investments in a number of assets and take advantage of *diversification* to reduce investment risk. In this paper we aim to provide a competitive portfolio management strategy exploiting RL.

A simple "divide-and-conquer" approach using RL for portfolio management can be divided into two steps. First, RL is run separately on each available asset

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1020–1031, 2007.

to obtain the Q-values of different actions; Second, asset weights are generated based on these Q-values. However, this approach is subject to two questions. First, the obtained weights may not be optimal in terms of profit as RL neglects the inter-relations between the returns of different assets. Second, whether it is good to apply RL directly on assets is still under question. This is because the prediction of future reward is important to RL's performance while it is well known that asset return is difficult to predict.

The independent factor model in finance [8, 9, 10, 11] may help to address the above two questions. In the independent factor model, the observed asset returns are believed to be linear mixtures of some hidden independent factors. Motivated by these works, we proposed to use RL on independent factors instead of on asset returns as in past works. As the factors are as independent as possible, the inter-relations between them are almost negligible. Hence we no longer need to consider the inter-relations. Also there have been many research works believing that the independent factors are more structured and regular, and consequently can be predicted better [12, 13] than asset returns. In this way using independent factors can be expected to strengthen the usefulness of RL in portfolio management.

Therefore, in this paper, we propose to do portfolio management by virtue of RL and the independent factor model. This proposed system is named independent factor RL (IF-RL) for portfolio management. The independent factors can be estimated by using independent component analysis (ICA) [14], a statistical technique for revealing hidden factors underlying the observed signals with a linear transformation. The system implementation consists of four steps. Firstly, ICA is used to construct independent factors from asset returns. Secondly, as the factors are almost independent from each other, RL is run on all factors in parallel to obtain investment suggestions on factors. Thirdly, portfolio optimization method is used to find factor weights that optimize specific objective function subject to investment suggestions from RL and general portfolio constraints. Lastly, optimal asset weights are obtained by converting optimal factor weights.

The rest of the paper is organized as follows. In Sect.2 the design of the proposed IF-RL system is described in detail. In Sect.3, another RL-based portfolio management system without utilizing independent factors is formulated for comparison with the proposed system in later experiments. Experimental results and analysis are provided in Sect.4 to compare the performances of IF-RL system and two alternative portfolio management systems. Finally Sect.5 discusses some future works and concludes.

2 Proposed System

In this section, we describe in detail the design of the proposed IF-RL system. Fig.1 shows the structure of the system which can be divide into two parts. The inner part within the dotted-line block is a RL-based portfolio management model consisting of a RL module and a portfolio optimization overlay. This

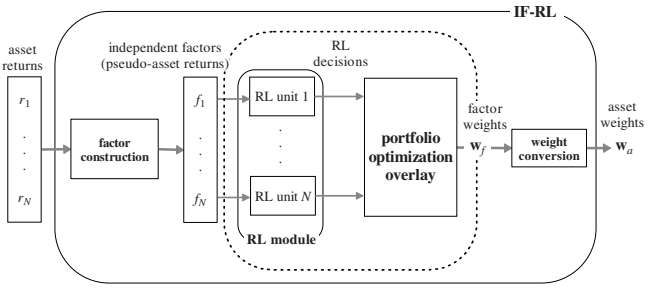


Fig. 1. The structure of IF-RL portfolio management system

model is designed to operate on multiple assets. The outer part is composed of factor construction module and weight conversion module which are in charge of transformation between assets and independent factors. In the following text, we elaborate the two parts respectively.

2.1 Factor Construction Module

Factor construction module extracts factors from asset returns. In this paper we adopt the FastICA algorithm [15] to extract independent components from returns.

Assume that we can invest in a market consisting of N risky assets and risk-free cash. At time t, let $\epsilon_i(t)$ be the return of asset i at t, which is defined as

$$\epsilon_i(t) = \frac{p_i(t+1) - p_i(t)}{p_i(t)}$$

where $p_i(t)$ is the price of asset i at t. The asset returns at t can be summarized with a vector $\epsilon(t) = (\epsilon_1(t), \ldots, \epsilon_N(t))^T$, and for $t = 1, \ldots, T$, a return matrix $\epsilon = (\epsilon(1), \ldots, \epsilon(T))$ can be formed where each row represents the historical returns of a single asset. In the independent factor model, the returns $\epsilon_1(t), \ldots, \epsilon_N(t)$ are assumed to be linear combinations of some independent factors. To recover the independent factors, ICA uses the linear transformation

$$\mathbf{f}(t) = \mathbf{B}\epsilon(t) \tag{1}$$

where $\mathbf{f}(t) = (f_1(t), f_2(t), \cdots, f_N(t))^T$ with $f_i(t)$ being the ith recovered factor at t[1], and the matrix $\mathbf{B}$ is the de-mixing matrix for $\epsilon(t)$.

With a proper de-mixing matrix $\mathbf{B}$, we can implement the factor construction module. In IF-RL system, we consider the factors as returns of some pseudo-assets [2].

[1] For simplicity, we assume that the number of factors is equal to that of the assets.

[2] In the rest of this paper, for simplicity, we use factor in the place of pseudo-asset unless noted otherwise.

2.2 Weight Conversion Module

The weight conversion module converts the optimal factor weights obtained from the inner part to corresponding asset weights. Let the asset weights and factor weights be $\mathbf{w}_a$ and $\mathbf{w}_f$ respectively. We have

$$\mathbf{w}_a^T \epsilon(t) = \mathbf{w}_f^T \mathbf{f}(t)$$

By substituting Eq.(1) into the above equation, we can have the relation between $\mathbf{w}_a$ and $\mathbf{w}_f$

$$\mathbf{w}_a = \mathbf{B}^T \mathbf{w}_f \tag{2}$$

Like asset weights, factor weights are also subject to some portfolio constraints when utilized in portfolio management task. The general portfolio constraints on asset weights can be stated as[3]

$$\sum_{i=1}^{N} w_{ai} \leq 1 \text{ and } \forall i = 1, \ldots, N \; w_{ai} \geq 0 \tag{3}$$

where w_{ai} is the asset weight on asset i. The sum of asset weights is set to be no bigger than 1 as there may be some capital allocated in cash. The equation can be rewritten in matrix form as

$$[1]_N^T \mathbf{w}_a \leq 1 \text{ and } \mathbf{w}_a \geq [0]_N$$

where $[1]_N$ and $[0]_N$ are respectively N-dimension vector of all 1's and all 0's. By utilizing Eq.(3) and Eq.(2), general portfolio constraints on factor weights can be specified as

$$[1]_N^T \mathbf{w}_f \leq 1 \text{ and } \mathbf{B}^T \mathbf{w}_f \geq [0]_N \tag{4}$$

These constraints will be used in portfolio optimization with respect to factor weights (see Sect.2.4).

2.3 RL Module

The RL module consists of many RL units, which structure is shown in Fig.2, with each unit operating on one factor. RL units are run in parallel, and output RL decisions on corresponding factors. In practice, it is common for investors to set constraints on the proportion of specific asset in the portfolio. Similarly, in this paper we interpret RL decisions as suggestions on degree of investment of factors, i.e. constraints on factor weights.

In each RL unit, we solve a single-asset allocation problem trading with one factor. The available actions are -1 and 1, representing respectively short and long position. We choose this action setting because we can see from Eq.(4) that factor weights may be negative, which indicates short-selling of the factors. At

[3] Please be noted that in this paper we assume non-negative asset weights to disallow short-selling.

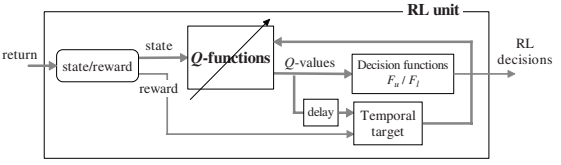


Fig. 2. The structure of RL unit

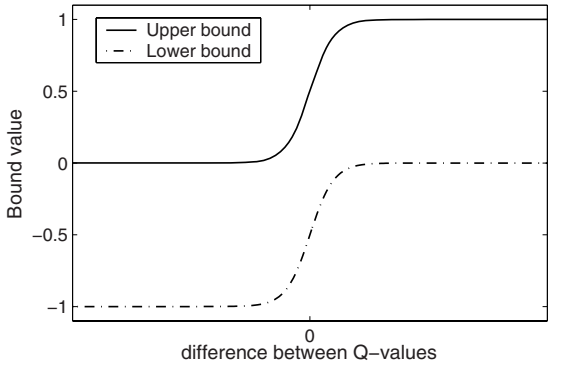


Fig. 3. Example of upper and lower bound on factor weight generated with RL

time t, for factor i, the state $s_{it} = (\$_{it}, k_{it})$ consists of two parts: $\$_{it}$ describes the market impact which is independent of investor's decision; while $k_{it} \in \{-1, 1\}$ represents the current investment position (short or long). Within each RL unit, we maintain Q-values of the binary actions. Q-values represent the expected future return of applying specific action at given state, and can be updated with the following formula during training:

$$Q_a(s) \leftarrow (1-\eta) Q_a(s) + \eta (r + \gamma \max (Q_{-1}(s'), Q_1(s')))$$

where $\langle s, a, r, s' \rangle$ is the observed tuple of current state, applied action, perceived reward and next state, η is learning rate, $0 \leq \gamma \leq 1$ is the discount factor in RL, and $Q_a(s)$ is estimated Q-value of applying action a at state s. In trading system, generally the reward can be represented as the capital gain subtracted by the transaction cost, i.e. at time t, for factor i, we have $r_{it} = g_{it} + c_{it}$, where g_{it} is the change of total capital during $[t, t+1]$ due to the price variations, and c_{it} the commission charge for traing at t, if applicable. For convenience, transaction cost is assumed non-positive to denote the *paid* charge. In the context of factor trading, the capital gain and transaction cost can be computed with

$$g_{it} = \log (1 + a_{it} \cdot f_i(t))$$
$$c_{it} = \log \left(1 - \delta \cdot \sum_{j=1}^{N} |b_{ij}| \cdot |k_{it} - a_{it}| \right)$$

where δ is transaction cost rate of asset trading.

At time t, for factor i, the optimal action a_{it}^* can be determined via

$$a_{it}^* = \text{sgn}\left(d\left(s_{it}\right)\right) \tag{5}$$

where $d\left(s_{it}\right) = Q_1\left(s_{it}\right) - Q_{-1}\left(s_{it}\right)$ is the difference between Q-values at s_{it}, and sgn() is the sign function. In RL module, we use two sigmoid-shape functions $F_u\left(\right)$ and $F_l\left(\right)$ (see Eq.(6)) to generate decisions on upper bound and lower bound of factor weight to control the weight from approaching boundary values of -1 and 1.

$$\begin{aligned} F_u\left(d\left(s_{it}\right)\right) &= \frac{1}{2}\left(1 + \tanh\left(N_u \cdot d\left(s_{it}\right)\right)\right) \\ F_l\left(d\left(s_{it}\right)\right) &= \frac{1}{2}\left(-1 + \tanh\left(N_l \cdot d\left(s_{it}\right)\right)\right) \end{aligned} \tag{6}$$

where N_u and N_l are respectively upper/lower bound parameter. The outputted RL decisions can be stated as

$$\forall i = 1, \ldots, N \quad F_l\left(d\left(s_{it}\right)\right) \leq w_{fi} \leq F_u\left(d\left(s_{it}\right)\right) \tag{7}$$

An example of upper and lower bound is shown in Fig.3, we can see from the figure that when RL prefers action 1 (or -1), the difference between Q-values is a positive (or negative) value, the upper and lower bound approaches to 1 and 0 (or 0 and -1) respectively with greater preference, i.e. bigger absolute value of difference between Q-values.

2.4 Portfolio Optimization Overlay

The portfolio optimization overlay can be implemented with various portfolio optimization methods. In this paper's experiments, we use mean-variance optimization (MVO) model, which was initialized by Markowitz in his landmark paper [16] and may be the most renowned portfolio optimization method. In MVO, the objective function to be maximized can be expressed as

$$U\left(\mathbf{w}_a\right) = \mathbf{w}_a^T \overline{\epsilon} - u\mathbf{w}_a^T \mathbf{\Sigma} \mathbf{w}_a$$

where $\overline{\epsilon}$ and $\mathbf{\Sigma}$ are respectively expected return and covariance matrix, $u \geq 0$ is the risk aversion. By utilizing Eq.(2), we can state the optimization problem in IF-RL's portfolio optimization overlay as maximizing the following objective function subject to constraints in Eq.(4) and Eq.(7)

$$U\left(\mathbf{w}_f\right) = \mathbf{w}_f^T \mathbf{B} \overline{\epsilon} - u\mathbf{w}_f^T \mathbf{B} \mathbf{\Sigma} \mathbf{B}^T \mathbf{w}_f$$

3 Another RL-Based Portfolio Management System

In this section, we formulate another RL-based portfolio management system as shown in Fig.4. We can find from the figure that this system has a structure similar to IF-RL system, except that RL is used directly on asset returns

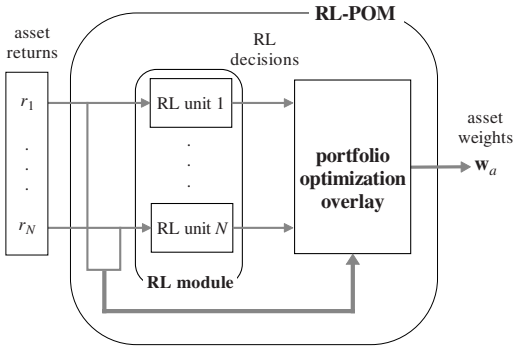


Fig. 4. The structure of RL-POM system

instead of independent factors. The system is named RL-POM as it combines RL module and portfolio optimization overlay. In the sysem, RL is run separately on inter-related asset returns, which may lead to suboptimal portfolio. Also the poor predictability of asset returns will limit the usefulness of RL in portfolio management. The RL-POM system will be used for comparison with the IF-RL system in experiments to investigate the advatange of applying RL on independent factors.

In the RL-POM system, the RL module is also composed of many RL units. RL units provide RL decisions on degree of investment of assets. Since the asset weight lies in the range $[0, 1]$, the available actions are set as 0 (invest in cash) and 1 (invest in risky asset). At time t, for asset i, let the Q-values of binary actions be $Q_0(s_{it})$ and $Q_1(s_{it})$ respectively, the optimal action a^*_{it} is

$$a^*_{it} = H(d(s_{it})) \tag{8}$$

where $d(s_{it}) = Q_1(s_{it}) - Q_0(s_{it})$ is the difference between Q-values at s_{it}, and $H()$ is heaviside step function. In RL-POM we only need to provide decision on upper bound of asset weights with

$$F_u(d(s_{it})) = \frac{1}{2}(1 + \tanh(N_u \cdot d(s_{it})))$$

The outputted RL decisions are

$$\forall i = 1, \ldots, N \quad w_{ai} \leq F_u(d(s_{it})) \tag{9}$$

The portfolio optimization overlay directly find the asset weights $\mathbf{w}_a$ that optimize specific objective function while subject to constraints in Eq.(3) and Eq.(9).

4 Experimental Results and Analysis

Experiment on real stock data in Hong Kong market is provided in this section to illustrate the performance of proposed IF-RL portfolio management system.

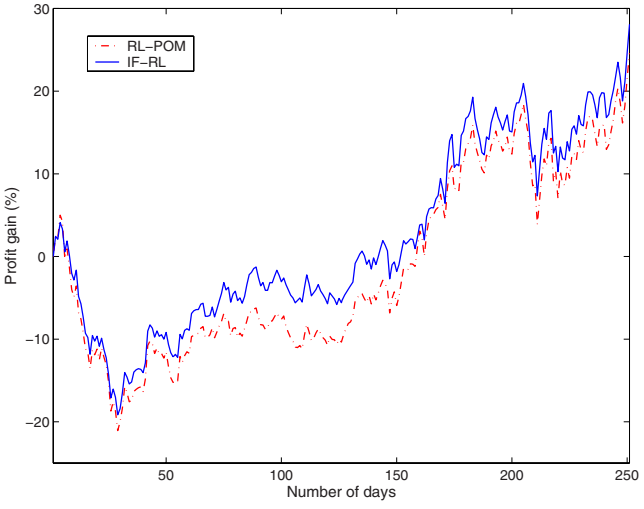


Fig. 5. Profit gains of the two portfolio management systems in the testing stage

The experiment is carried out by investing in a portfolio of 8 stocks[4]. The experimental data consists of 1000 data points from April 22, 2003 to May 4, 2007. The first 750 data points are used as training data set, whereas the remaining 250 days compose the testing stage. The transaction cost rate δ is set as 0.5%.

4.1 Experimental Results

In the experiment, we compare the performances of two portfolio management systems, i.e. RL-POM system and IF-RL system, described respectively in previous two sections. For both systems, MVO is chosen to implement portfolio optimization overlay with risk aversion $u = 1$, and the upper/lower bound parameter N_u and N_l are both set as 100.

For both systems, the 250-day testing stage is divided into 5 segments, and asset weights are rebalanced at the beginning of each segment. In the testing stage, the tested segment will be added to the training data set before moving forward to next segment; and the RL modules in RL-POM and IFRL system continue to train their trading policies with newly-observed data. This mechanism enables the systems adaptive to the changes in the dynamic market.

Fig.4 shows the profit gains of the three systems in the testing stage. The proposed IF-RL system can be found to outperform the RL-POM systems in terms of profitability. We also notice that IF-RL system can control the loss better when asset prices decrease, e.g. around day 50 to 60.

[4] The 8 stocks are all constituent stocks of Hang Seng Index in Hong Kong market. They are 0002-0003.HK, 0005-0006.HK, 0011.HK, 0013.HK, and 0016-0017.HK.

Table 1. Performances of the three portfolio management systems

system name	profit (%)	mean return	risk	Sharpe Ratio	dod
RL-POM	24.59	9.92e-4	2.25e-4	6.61e-2	0.4978
IFRL	28.10	1.10e-3	2.12e-4	7.54e-2	0.5976

Table 1 provides more performance statistics including mean return, risk, Sharpe Ratio, etc. All the performance statistics are measured on the testing stage, and risk is defined as variance of the trading return. In the table, dod stands for *degree of diversification* [17]. This measure tells how well the system diversifies its investments. It is computed with

$$\text{dod} = \frac{1}{N} \sum_{m=1}^{N} \mathbf{w}_m^T \left[[1]_N - \mathbf{w}_m \right]$$

where N is the number of segments, and $\mathbf{w}_m$ is the asset weights determined at the beginning of segment m.

From the table we can have two observations. First, the IF-RL system achieves a higher mean return, lower risk and better Sharpe Ratio when compared to the RL-POM system. These results can provide some positive evidences for the conjecture that independent factors may have better predictability than asset returns. Second, in terms of dod, the IF-RL system can achieve a more diversified portfolio than the RL-POM system does.

4.2 Analysis on Portfolio Formation

Besides the performance measures discussed above, we are also interested in the optimal asset weights found by the three systems. In Fig.6 we show the asset weights of the two systems determined at the beginning of the 5 segments. Among all the 8 assets, there are 7 assets selected by at least one system during the testing stage. We use a clustering of 7 bars to represent asset weights, where the bar height is equal to the weight on the corresponding asset. For those bars with non-zero height, asset indexes are marked on top of them.

It can be observed from Fig.6 that asset 8 is consistently selected by both systems as a major component. This can be contributed to this asset's significant profitability when compared to other 7 assets. Despite this similarity in constructing portfolio, the portfolios found by the IF-RL system are still different from those found by RL-POM system in terms of the minor portfolio components. This may be the reason why IF-RL system can achieve better performance than the RL-POM system. To further demonstrate this, we pick some example segments to show how different asset selections by IF-RL and RL-POM system lead to different performances.

In segment 2, while both systems choose to invest in asset 3 and 8, IF-RL system also invests in asset 1 and prefers it to asset 3. Fig.7(a) shows the normalized

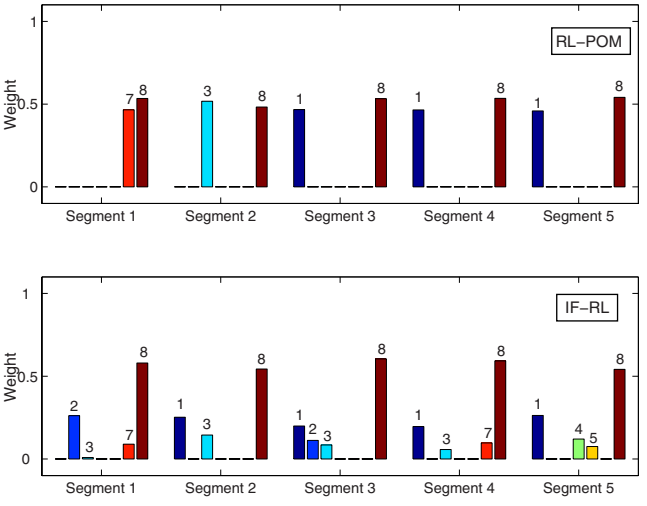
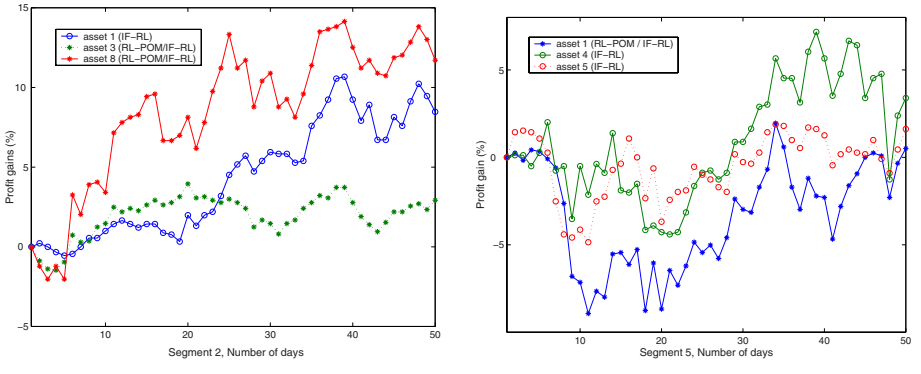


Fig. 6. Optimal asset weights found by two portfolio management systems for five segments



(a) Segment 2: asset 1 (IF-RL), 3 and 8 (RL-POM and IF-RL)

(b) Segment 5: asset 1 (RL-POM and IF-RL), 4 and 5 (IF-RL)

Fig. 7. Normalized prices of assets selected by different systems in two example segments

prices[5] of asset 1, 3 and 8 during segment 2. We can see that asset 1 outperforms asset 3 in terms of profit, indicating a better choice of IF-RL system. The opposite trends of asset 1 and 3 in the middle part of the segment show that adding asset 1 to portfolio can effectively reduce the portfolio risk. In segment 5, both system invest similar amount of capital in asset 8, but IF-RL system chooses to

[5] Here the asset prices are normalized so that prices at the beginning of the segment are 1, the same normalization is also used for segment 5.

diversify the remaining capital in asset 1, 4 and 5 while RL-POM invests only in asset 1. Fig.7(b) depicts the normalized prices of asset 1, 4 and 5 in segment 5. We can see that asset 4 and 5 outperform asset 1, which indicates that the diversification decision of IF-RL system is correct.

The analysis of portfolio formation shows that with the assistance of independent factors, IF-RL system can take advantage of the better predictability of factors and find better-performing portfolios.

5 Conclusion and Future Work

In this paper, we propose Independent Factor RL (IF-RL) system for portfolio management. With the assistance of independent factors, we can operate RL on all factors in parallel, which enables an efficient system structure. Also, combining independent factors with RL can take advantages of both techniques: RL has good forecasting power, while independent factors are believed to have better predictability than asset returns. Experimental results on real stock data in Hong Kong market show that IF-RL system achieves better trading performance than the comparative MVO model and RL-POM system. Analysis on portfolio formation shows that IF-RL system attempt to find better-performing portfolio that is different in formation from the portfolios found by other two systems, thus demonstrating the usefulness of independent factors.

Future work may include using other ICA techniques to extract independent factors, as well as applying IF-RL system on more data sets with more optimization criterions such as those related to downside risk.

References

[1] Kaelbling, L.P., Littman, M.L., Moore, A.P.: Reinforcement learning: A survey. Journal of Artificial Intelligence 4, 237–285 (1996)
[2] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
[3] Roy, R.V.: Neuro-dynamic programming and reinforcement learning for finance. In: Proc. Computational Finance (1999)
[4] Neuneier, R.: Optimal asset allocation using adaptive dynamic programming. In: Touretzky, D.S., Mozer, M., Hasselmo, M.E. (eds.) NIPS, pp. 952–958. MIT Press, Cambridge (1995)
[5] Neuneier, R.: Enhancing q-learning for optimal asset allocation. In: Jordan, M.I., Kearns, M.J., Solla, S.A. (eds.) NIPS, pp. 936–942. MIT Press, Cambridge (1997)
[6] Ormoneit, D., Glynn, P.: Kernel-based reinforcement learning in average-cost problems: An application to optimal portfolio choice. In: NIPS, pp. 1068–1074 (2000)
[7] Dempster, M.A.H., Leemans, V.: An automated fx trading system using adaptive reinforcement learning. Expert systems with applications: special issue on financial engineering 30, 534–552 (2006)
[8] Back, A.D.: A first application of independent component analysis to extracting structure from stock returns. International Journal of Neural Systems 8(4), 473–484 (1997)

[9] Kiviluoto, K., Oja, E.: Independent component analysis for parallel financial time series. In: Proc. ICONIP 1998, Tokyo, Japan, vol. 2, pp. 895–898 (1998)
[10] Cha, S.M., Chan, L.W.: Applying independent component analysis to factor model in finance. In: Proc. of IDEAL, pp. 538–544 (2000)
[11] Chan, L.W., Cha, S.M.: Selection of independent factor model in finance. In: 3rd International Conference on Independent Component Analysis and blind Signal Separation, San Diego, California, USA (December 2001)
[12] Pawelzik, K., Müller, K.R., Kohlmorgen, J.: Prediction of mixtures. In: Vorbrüggen, J.C., von Seelen, W., Sendhoff, B. (eds.) ICANN 96. LNCS, vol. 1112, pp. 127–132. Springer, Heidelberg (1996)
[13] Malaroiu, S., Kimmo, K., Oja, E.: Time series prediction with independent component analysis. In: Proc. Int. Conf. on Advanced Investment Technology, Gold Coast, Australia (2000)
[14] Hyvärinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. John Wiley & Sons, Inc, Chichester (2001)
[15] Hyvärinen, A.: Fast and robust fixed-point algorithms for independent component analysis. IEEE Transactions on Neural Networks 10(3), 626–634 (1999)
[16] Markowitz, H.: Portfolio selection. Journal of Finance 7(1), 77–91 (1952)
[17] Hung, K.K., Cheung, Y.M., Xu, L.: An extended asld trading system to enhance portfolio management. IEEE Transactions on Neural Networks 14(2), 413–425 (2003)

Discrete Time Portfolio Selection with Lévy Processes

Cesarino Bertini, Sergio Ortobelli Lozza, and Alessandro Staino

University of Bergamo, Department MSIA, Via dei Caniana, 2
24127 Bergamo, Italy

Abstract. This paper analyzes discrete time portfolio selection models with Lévy processes. We first implement portfolio models under the hypotheses the vector of log-returns follow or a multivariate Variance Gamma model or a Multivariate Normal Inverse Gaussian model or a Brownian Motion. In particular, we propose an ex-ante and an ex-post empirical comparisons by the point of view of different investors. Thus, we compare portfolio strategies considering different term structure scenarios and different distributional assumptions when unlimited short sales are allowed.

Keywords: Subordinated Lévy models, term structure, expected utility, portfolio strategies.

1 Introduction

In this paper, we model the returns as a multidimensional time-changed Brownian motion where the subordinator follows or an Inverse Gaussian process or a Gamma process. Under these different distributional hypotheses we compare the portfolio strategies with the assumption that the log-returns follow a Brownian motion.

The literature in the multi-period portfolio selection has been dominated by the results of maximizing expected utility functions of terminal wealth and/or multi-period consumption. Differently from classic multi-period portfolio selection approaches, we consider mean-variance analysis alternative to that proposed by Li and Ng's (2000) by giving a mean-dispersion formulation of the optimal dynamic strategies. Moreover we also discuss a mean, variance, skewness and kurtosis extension of the original multi-period portfolio selection problem. These alternative multi-period approaches analyze portfolio selection, taking into consideration the admissible optimal portfolio choices when the log-returns follow a Lévy process. This analysis differs from other studies that assume Lévy processes with very heavy tails (see Rachev and Mittnik (2000), Ortobelli et al. (2004)), since we consider Lévy processes with semi-heavy tails. In order to compare the dynamic strategies under the different distributional assumptions, we analyze two investment allocation problems. The primary contribution of this empirical comparison is the analysis of the impact of distributional assumptions and different term structures on the multi-period asset allocation decisions. Thus, we propose a performance comparison among different multi-period mean-variance approach based on different Lévy processes and taking into consideration three different implicit term structures. For this purpose we discuss

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1032–1041, 2007.

the optimal allocation obtained by different risk averse investors with different risk aversion coefficients. We determine the multi-period optimal choices given by the minimization of the variance for different levels of final wealth average. Each investor, characterized by his/her utility function, will prefer the mean-variance model which maximizes his/her expected utility on the efficient frontier. Thus the portfolio policies obtained with this methodology represent the optimal investors' choices of the different approaches.

In Section 2, we introduce dynamic portfolio selection under the different multivariate distributional hypotheses. In Section 3 we propose a comparison of optimal portfolio strategies. Section 4, briefly summarizes the results.

2 Discrete Time Portfolio Selection with Subordinated Lévy Processes

In this section we deal the dynamic portfolio selection problem among N+1 assets: N are risky assets and the (N+1)-th is risk free. We introduce portfolio selection models based on different assumptions of log-return distributions. In particular, we consider the Normal Inverse Gaussian process (NIG) and the Variance-Gamma one (VG) which are Lévy processes with semi heavy tails as suggested by Staino et al. (2007). These processes can be seen as subordinated Lévy processes where the subordinators are respectively the Inverse Gaussian process and the Gamma process.

Suppose that in the market the vector of risky assets has log-returns $\mathbf{X}_t = [X_t^{(1)},...,X_t^{(N)}]'$ distributed as:

$$\mathbf{X}_t = \mathbf{s}t + \boldsymbol{\mu} Z_t + \mathbf{Q}^{1/2} W_{Z_t}^{(N)}, \tag{1}$$

where $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_N]'$, $\mathbf{s} = [s_1, s_2, \ldots, s_N]'$, Z_t is the positive Lévy process subordinator, $\mathbf{Q} = \left[\sigma_{ij}^2\right]$ is a fixed definite positive variance-covariance matrix (i.e., $\sigma_{ij}^2 = \sigma_{ii}\sigma_{jj}\rho_{ij}$ where ρ_{ij} is the correlation between the conditional i-th component of $\mathbf{X}_t / Z_t$ and its conditional j-th component) and $W_t^{(N)}$ is a N-dimensional standard Brownian motion (i.e., $\mathbf{Q}^{1/2} W_{Z_t}^{(N)} = \sqrt{Z_t}\,\mathbf{Q}^{1/2}\mathbf{Y}$ where $\mathbf{Y}$ is a standard N-dimensional Gaussian independent of Z_t). Under the above distributional hypotheses we approximate the log-return of the portfolio with the portfolio of log-returns, that is the convex combination of the log-returns:

$$X_t^{(w)} = \mathbf{w}\mathbf{X}_t = \mathbf{w}\mathbf{s}t + (\mathbf{w}\boldsymbol{\mu})Z_t + \sqrt{\mathbf{w}\mathbf{Q}\mathbf{w}'}W_{Z_t}, \tag{2}$$

where $\mathbf{w} = [w_1, \ldots, w_N]$ is the vector of the weights invested in the risky assets, W_t is a 1-dimensional standard Brownian motion. At this point we will assume, that the subordinator Z_t is modeled either as an Inverse Gaussian process $Z_1 \sim \mathbf{IG}(1, b)$ or as a Gamma process $Z_1 \sim \mathbf{Gamma}\left(\frac{1}{\nu}, \frac{1}{\nu}\right)$. An Inverse Gaussian process $X^{(IG)} = \left\{X_t^{(IG)}\right\}_{t\geq 0}$ assumes that any random variable $X_t^{(IG)}$ admits the following density function:

$$f_{IG}(x;ta,b)=\frac{ta}{\sqrt{2\pi}}\exp(tab)x^{-3/2}\exp\left(-\frac{1}{2}(t^2a^2x^{-1}+b^2x)\right)1_{[x>0]},$$

that is defined as Inverse Gaussian distribution $IG(ta,b)$ where *a, b* are positive. A Gamma process $X^{(Gamma)}=\left\{X_t^{(Gamma)}\right\}_{t\geq 0}$ assumes that any random variable $X_t^{(Gamma)}$ admits the following density function:

$$f_{Gamma}(x;t\,a,b)=\frac{b^{ta}}{\Gamma(ta)}x^{ta-1}\exp(-xb)1_{[x>0]},$$

that is defined as Gamma distribution $Gamma(ta,b)$ where *a, b* are positive.

NIG Processes. When Z_t follows an Inverse Gaussian process (i.e., $Z_1 \sim \mathtt{IG}(1,b)$), then, the *i*-th log-return at time t=1 follows a NIG process $\mathtt{NIG}(\alpha_i,\beta_i,\delta_i,s_i)$ where the parameters are given by: $\alpha_i=\sqrt{(b/\delta_i)^2+\beta_i^2}$, $\beta_i=\mu_i/\delta_i^2\in(-\alpha_i,\alpha_i)$, $s_i\in R$ and $\delta_i=\sigma_{ii}>0$. Thus, the portfolio (2) follows a $\mathtt{NIG}(\alpha_w,\beta_w,\delta_w,s_w)$ process whose parameters are: $s_w=\mathbf{ws}$, $\alpha_w=\sqrt{\left(\frac{b}{\delta_w}\right)^2+\beta_w^2}$, $\beta_w=\frac{\mathbf{w\mu}}{\mathbf{wQw'}}$, $\delta_w=\sqrt{\mathbf{wQw'}}$. In order to estimate all these parameters, we follow the MLE procedure suggested by Staino et al. (2007).

Variance-Gamma Processes. When Z_t follows a Gamma process (i.e., at time t=1 $Z_1 \sim \mathtt{Gamma}\left(\frac{1}{\nu},\frac{1}{\nu}\right)$), then the log-return of the *i*-th asset at time t=1 follows a Variance-Gamma process with parameters $s_i,\mu_i\in R$, and $\nu,\sigma_{ii}>0$ (i.e., $X_1^{(i)}\sim \mathtt{VG}(\mu_i,\sigma_i,\nu,s_i)$). Analogously, the portfolio (2) follows a Variance-Gamma process with parameters $s_w=\mathbf{ws}$, $\mu_w=\mathbf{w\mu}$, $\sigma_w=\sqrt{\mathbf{wQw'}}$ and ν. As for the NIG process we estimate all these parameters following the MLE procedure suggested by Staino et al. (2007). In portfolio theory, it has been widely used a Brownian Motion to model the vector of log-returns distribution. Under this assumption the portfolio follows a Brownian Motion $X^{(BM)w}=\left\{X_t^{(BM)w}\right\}_{t\geq 0}$ process, that is the portfolio of log returns at time t is normal distributed with mean $\mathbf{w\mu}t$ and standard deviation $\sqrt{t\mathbf{wQw'}}$. In the next subsection we describe the portfolio selection problem under the different distributional assumptions.

2.1 The Discrete Time Portfolio Selection Problem

Suppose an investor has a temporal horizon t_T and he recalibrates its portfolio T times at some intermediate date, say $t=t_0,\ldots,t_{T-1}$ (where $t_0=0$). Since Lévy processes have independent and stationary increments the distribution of the random vector of log-returns on the period $(t_j,t_{j+1}]$ (i.e., $\mathbf{X}_{t_{j+1}}-\mathbf{X}_{t_j}$) is the same of $\mathbf{X}_{t_{j+1}-t_j}=[X^{(1)}_{t_{j+1}-t_j},\ldots,X^{(N)}_{t_{j+1}-t_j}]'$. Considering that log-returns represent a good

approximation of returns when $t_{j+1}-t_j$ is little enough, we assume that $\max_{j=0,\ldots,T-1}\left(t_{j+1}-t_j\right)$ is less or equal than one month and we use $\mathbf{Y}_{t_j} := \mathbf{X}_{t_{j+1}} - \mathbf{X}_{t_j} = [Y_{1,t_j},\ldots,Y_{N,t_j}]'$ to estimate the vector of returns on the period $(t_j, t_{j+1}]$. Suppose the deterministic variable r_{0,t_j} represents the return on the period $(t_j, t_{j+1}]$ of the risky-free asset, x_{i,t_j} the amount invested at time t_j in the *i*-th risky asset, and x_{0,t_j} the amount invested at time t_j in the risky-free asset. Then the investor's wealth at time t_{k+1} is given by:

$$\mathrm{W}_{t_{k+1}} = \sum_{i=0}^{N} x_{i,t_k}(1+Y_{i,t_k}) = (1+r_{0,t_k})\mathrm{W}_{t_k} + \mathbf{x}_{t_k}\mathbf{P}_{t_k}, \tag{3}$$

where $\mathbf{x}_{t_k} = [x_{1,t_k},\ldots,x_{N,t_k}]$, $\mathbf{P}_{t_k} = [P_{1,t_k},\ldots,P_{N,t_k}]'$ is the vector of excess returns $P_{i,t_k} = Y_{i,t_k} - r_{0,t_k}$. Thus, the final wealth is given by:

$$\mathrm{W}_{t_T} = \mathrm{W}_0\prod_{k=0}^{T-1}(1+r_{0,t_k}) + \sum_{i=0}^{T-2}\mathbf{x}_{t_i}\mathbf{P}_{t_i}\prod_{k=i+1}^{T-1}(1+r_{0,t_k}) + \mathbf{x}_{t_{T-1}}\mathbf{P}_{T-1}, \tag{4}$$

where the initial wealth $\mathrm{W}_0 = \sum_{i=0}^{N} x_{i,0}$ is known. Assume that the amounts $\mathbf{x}_{t_j} = [x_{1,t_j},\ldots,x_{N,t_j}]$ are deterministic variables, whilst the amount invested in the risky-free asset is the random variable $x_{0,t_j} = W_{t_j} - \mathbf{x}_{t_j}\mathbf{e}$, where $\mathbf{e} = [1,\ldots,1]'$. Under these assumptions the mean, the variance the skewness and kurtosis of the final wealth are respectively

$$E(\mathrm{W}_{t_T}) = \mathrm{W}_0 B_0 + \sum_{i=0}^{T-1} E(\mathbf{x}_{t_i}\mathbf{P}_{t_i})B_{i+1} \tag{5a}$$

$$\mathbf{Variance}\left(W_T\right) = \sigma^2(W_{t_T}) = \textstyle\sum_{i=0}^{T-1}\left(\mathbf{x_{t_i}}\mathbf{Q_{t_i}}\mathbf{x'_{t_i}}\right)B_{i+1}^2 \tag{5b}$$

$$\mathbf{Sk}\left(\mathrm{W}_{t_T}\right) = \frac{\sum_{i=0}^{T-1} E\left((\mathbf{x}_{t_i}\mathbf{P}_{t_i} - E(\mathbf{x}_{t_i}\mathbf{P}_{t_i}))^3\right)B_{i+1}^3}{\sigma^3(W_{t_T})} \tag{5c}$$

$$\mathbf{Ku}\left(\mathrm{W}_{t_T}\right) = \frac{6\sum_{i=0}^{T-1}\sum_{j=i+1}^{T-1} B_{j+1}^2 B_{i+1}^2 \mathbf{x_{t_i}}\mathbf{Q_{t_i}}\mathbf{x'_{t_i}}\left(\mathbf{x_{t_j}}\mathbf{Q_{t_j}}\mathbf{x'_{t_j}}\right) + \sum_{i=0}^{T-1} E\left((\mathbf{x}_{t_i}\mathbf{P}_{t_i} - E(\mathbf{x}_{t_i}\mathbf{P}_{t_i}))^4\right)B_{i+1}^4}{\sigma^4(W_{t_T})} \tag{5d}$$

where $B_T = 1$, $B_k = \prod_{j=k}^{T-1}(1+r_{0,t_j})$. Therefore, if we want to select the optimal portfolio strategies that solve the mean-variance problem:

$$\begin{cases} \min\limits_{\mathbf{x}_{t_0},\ldots,\mathbf{x}_{t_{T-1}}} \mathbf{Variance}[\mathrm{W}_{t_T}] \\ \mathbf{s.t.}\ E[\mathrm{W}_{t_T}] = m \end{cases},$$

we can use the closed form solutions determined by Ortobelli et al. (2004). These solutions for Lévy subordinated processes are given by:

$$\mathbf{x}'_{t_k} = \frac{m - \mathrm{W}_0 B_0}{B_{k+1}\sum_{j=0}^{T-1} E(\mathbf{P}_{t_j})'\mathbf{Q}_{t_j}^{-1} E(\mathbf{P}_{t_j})} \mathbf{Q}_{t_k}^{-1} E(\mathbf{P}_{t_k}) \quad k = 0,\ldots,T-1 \tag{6}$$

where the components of the matrix $\mathbf{Q}_{t_k} = [q_{ij,t_k}]$, ($k$=0,…,$T$-1), are $q_{ij,t_k} = \mathbf{Cov}(X^{(i)}_{t_{k+1}-t_k}, X^{(j)}_{t_{k+1}-t_k})$. The optimal wealth invested in the riskless asset at time $t_0 = 0$ is the deterministic quantity $x_{0,0} = \mathrm{W}_0 - \mathbf{x}_{t_0}\mathbf{e}$, while at time t_j it is given by the random variable $x_{0,t_j} = \mathrm{W}_{t_j} - \mathbf{x}_{t_j}\mathbf{e}$, where W_{t_j} is formulated in equation (3). Observe that the covariance q_{ij,t_k} among components of the vector $\mathbf{X}_{t_{j+1}-t_j} = \mathbf{s}(t_{j+1}-t_j) + \boldsymbol{\mu} Z_{t_{j+1}-t_j} + \mathbf{Q}^{1/2} W^{(N)}_{Z_{t_{j+1}-t_j}}$ is given by

$$q_{ij,t_k} = \sigma_{ij}^2 E(Z_{t_{k+1}-t_k}) + \mu_i \mu_j \mathbf{Variance}[Z_{t_{k+1}-t_k}],$$

where $\sigma_{ij}^2 = \sigma_{ii}\sigma_{jj}\rho_{ij}$ are the components of matrix $\mathbf{Q} = \left[\sigma_{ij}^2\right]$ (see, among others, Cont and Tankov (2004)). So, for example, in the case the vector of log-returns $\mathbf{X}_t$ follows a NIG process we can rewrite the formulas (5) of mean, variance, skewness and kurtosis of final wealth:

$$E(\mathbf{x}_{t_i}\mathbf{P}_{t_i}) = (t_{i+1}-t_i)\left(b^{-1}\mathbf{x}_{t_i}\boldsymbol{\mu} + \mathbf{x}_{t_i}\mathbf{s}\right) - \mathbf{x}_{t_i}\mathbf{e}r_{0,t_i}$$

$$q_{ij,t_k} = \mathbf{Cov}(X^{(i)}_{t_{k+1}-t_k}, X^{(j)}_{t_{k+1}-t_k}) = \delta_i\delta_j\rho_{ij}E(I_{t_{k+1}-t_k}) +$$
$$+\beta_i\beta_j\delta_i^2\delta_j^2\mathbf{Variance}[I_{t_{k+1}-t_k}] = \frac{\delta_i\delta_j\rho_{ij}}{b}(t_{k+1}-t_k) + \frac{\beta_i\beta_j\delta_i^2\delta_j^2}{b^3}(t_{k+1}-t_k)$$

$$\mathbf{Sk}\left(\mathrm{W}_{t_T}\right) = \frac{3\sum_{i=0}^{T-1} B_{i+1}^3 (t_{i+1}-t_i)\mathbf{x}_{t_i}\boldsymbol{\mu}\left(b^2\mathbf{x}_{t_i}\mathbf{Q}_{t_i}\mathbf{x}'_{t_i} + \left(\mathbf{x}_{t_i}\boldsymbol{\mu}\right)^2\right)}{\sigma^3(W_{t_T})b^5}$$

$$\mathbf{Ku}\left(\mathrm{W}_{t_T}\right) = \frac{6\sum_{i=0}^{T-1}\sum_{j=i+1}^{T-1} B_{j+1}^2 B_{i+1}^2 \mathbf{x}_{\mathbf{t_i}}\mathbf{Q}_{\mathbf{t_i}}\mathbf{x}'_{\mathbf{t_i}}\left(\mathbf{x}_{\mathbf{t_j}}\mathbf{Q}_{\mathbf{t_j}}\mathbf{x}'_{\mathbf{t_j}}\right)}{\sigma^4(W_{t_T})} +$$
$$+\frac{3\sum_{i=0}^{T-1} B_{i+1}^4 (t_{i+1}-t_i)^2\left(b^2\mathbf{x}_{t_i}\mathbf{Q}_{t_i}\mathbf{x}'_{t_i} + \left(\mathbf{x}_{t_i}\boldsymbol{\mu}\right)^2\right)^2}{\sigma^4(W_{t_T})b^6} +$$

$$+\frac{3\sum_{i=0}^{T-1}B_{i+1}^4\left(t_{i+1}-t_i\right)\left(b^2\mathbf{x}_{t_i}\mathbf{Q}_{t_i}\mathbf{x}'_{t_i}+5\left(\mathbf{x}_{t_i}\boldsymbol{\mu}\right)^2\right)\left(b^2\mathbf{x}_{t_i}\mathbf{Q}_{t_i}\mathbf{x}'_{t_i}+\left(\mathbf{x}_{t_i}\boldsymbol{\mu}\right)^2\right)}{\sigma^4(W_{t_T})b^7}$$

Instead, if $\mathbf{X}_t$ follows a Variance-Gamma process these formulas become:

$$E\left(\mathbf{x}_{t_i}\mathbf{P}_{t_i}\right)=\left(t_{i+1}-t_i\right)\left(\mathbf{x}_{t_i}\boldsymbol{\mu}+\mathbf{x}_{t_i}\mathbf{s}\right)-\mathbf{x}_{t_i}\mathbf{e}r_{0,t_i}$$

$$q_{ij,t_k}=\mathbf{Cov}(X^{(i)}_{t_{k+1}-t_k},X^{(j)}_{t_{k+1}-t_k})=\sigma_{ii}\sigma_{jj}\rho_{ij}E(Z_{t_{k+1}-t_k})+\mu_i\mu_j\mathbf{Variance}[Z_{t_{k+1}-t_k}]=$$
$$=\sigma_{ii}\sigma_{jj}\rho_{ij}(t_{k+1}-t_k)+v\mu_i\mu_j(t_{k+1}-t_k).$$

$$\mathbf{Sk}\left(\mathrm{W}_{t_T}\right)=\frac{\sum_{i=0}^{T-1}\left(\left(t_{i+1}-t_i\right)v\mathbf{x}_{t_i}\boldsymbol{\mu}\left(3\mathbf{x}_{t_i}\mathbf{Q}_{t_i}\mathbf{x}'_{t_i}+2v\left(\mathbf{x}_{t_i}\boldsymbol{\mu}\right)^2\right)\right)B_{i+1}^3}{\sigma^3(W_{t_T})}$$

$$\mathbf{Ku}\left(\mathrm{W}_{t_T}\right)=\frac{6\sum_{i=0}^{T-1}\sum_{j=i+1}^{T-1}B_{j+1}^2B_{i+1}^2\mathbf{x}_{\mathbf{t_i}}\mathbf{Q}_{\mathbf{t_i}}\mathbf{x}'_{\mathbf{t_i}}\left(\mathbf{x}_{\mathbf{t}_j}\mathbf{Q}_{t_j}\mathbf{x}'_{\mathbf{t}_j}\right)}{\sigma^4(W_{t_T})}-\frac{\sum_{i=0}^{T-1}3B_{i+1}^4\left(v\left(\mathbf{x}_{\mathbf{t_i}}\mathbf{Q}_{\mathbf{t_i}}\mathbf{x}'_{\mathbf{t_i}}\right)^2\left(t_{i+1}-t_i\right)\right)}{\sigma^4(W_{t_T})}+$$
$$+\frac{\sum_{i=0}^{T-1}3B_{i+1}^4\left(\left(1+2v/\left(t_{i+1}-t_i\right)\right)\left(\mathbf{x}_{\mathbf{t_i}}\mathbf{Q}_{\mathbf{t_i}}\mathbf{x}'_{\mathbf{t_i}}\left(t_{i+1}-t_i\right)+v\left(\mathbf{x}_{t_i}\boldsymbol{\mu}\right)^2\left(t_{i+1}-t_i\right)\right)^2\right)}{\sigma^4(W_{t_T})}.$$

Clearly, a more realistic portfolio selection problem should consider the investor's preference for skewness (see, among others, Ortobelli (2001)). Thus under the above distributional assumptions and under institutional restrictions of the market (such as no short sales and limited liability), all risk-averse investors optimize their portfolio choosing the solutions of the following constrained optimization problem:

$$\min_{\mathbf{x}_{t_0},\ldots,\mathbf{x}_{t_{T-1}}}\ \mathbf{Variance}[\mathrm{W}_{t_T}]$$
$$\mathbf{s.t.}\ E[\mathrm{W}_{t_T}]\geq m;\ \mathbf{Sk}\left(\mathrm{W}_{t_T}\right)\geq q_1;\ \mathbf{Ku}\left(\mathrm{W}_{t_T}\right)\leq q_2;x_{i,t_j}\geq 0,i=1,\ldots,N,j=0,\ldots,T-1$$

for some mean m skewness q_1 and kurtosis q_2. This problem has not generally closed form solution. However using arguments similar to those proposed by Athayde, and Flôres (2004) based on a tensorial notation for the higher moments we can give an implicit analytical solution when unlimited short sales are allowed.

3 A Comparison Among Lévy Dynamic Strategies

In this section we examine the performances of Lévy processes approaches and we compare the Gaussian and Lévy non-Gaussian dynamic portfolio choice strategies when short sales are allowed. Since we work in mean variance framework, we do not value the effects of skewness a kurtosis that will be object of future researches. First, we analyze the optimal dynamic strategies during a period of five months, among the

riskless return and 5 index-monthly returns from 04/10/1992 - 12/31/2005 (Down Jones Composite 65, Down Jones Industrials, Down Jones Utilities, S&P 500 and S&P 100). We start with a riskless of 0.3884% and we examine the different allocation considering three different implicit term structures. Table 1 describes the implicit term structures that we will use in this comparison.

Table 1. Term structures

	t_0	t_1	t_2	t_3	t_4
term 1	0.3884%	0.3984%	0.4084%	0.4184%	0.4284%
term 2	0.3884%	0.3884%	0.3884%	0.3884%	0.3884%
term 3	0.3884%	0.3784%	0.3684%	0.3584%	0.3484%

In particular, we approximate optimal solutions to the utility functional:

$$\max_{\{x_{tj}\}_{j=0,1,\ldots,T-1}} E\left(1-\exp\left(-\frac{1}{a}W_T\right)\right) \tag{7}$$

where a (we use $a=0.5,1,1.5,2$) is an indicator of the risk tolerance and W_T is defined by formula (4). Secondly, we consider the utility function

$$u(x)=\begin{cases} cx-\frac{1}{c}x^2 & \text{if } x<\frac{c^2}{2} \\ \ln(x)+\left(\frac{c^3}{4}-\ln\left(\frac{c^2}{2}\right)\right) & \text{if } x\geq\frac{c^2}{2} \end{cases} \tag{8}$$

where for $x<c^2/2$ we have a quadratic utility function and for $x\geq c^2/2$ a logarithm utility function (we use $c=1,2,3,4,5$). Thus, we are interested in finding optimal solutions to the functional

$$\max_{\{x_{tj}\}_{j=0,1,\ldots,T-1}} E\left(u\left(W_T\right)\right) \tag{9}$$

Clearly we could obtain close form solutions to optimization problem (7) and (9) using arguments on the moments and on the Laplace transform. However, since we want to value the impact of different distributional assumptions in a mean-variance framework we will approximate formulas (7) and (9) using the historical observations of the final wealth valued for the optimal mean variance portfolios. In particular, we use the same algorithm proposed by Ortobelli et al. (2004) in order to compare the different models. Observed that transaction costs are not modeled in formula (6), however Staino et al. (2007) have shown there are not very big differences among portfolio choices with and without constant proportional transaction costs. Therefore, we do not consider transaction costs in this analysis. Thus, first we consider the optimal strategies (6) for different levels of the mean. Then, we select the optimal portfolio strategies on the efficient frontiers which are solutions of problem (7) and

(9) for different coefficients a and c. Therefore starting by an initial wealth $W_0 = 1$ we compute for every multi-period efficient frontier:

$$\max_{\{x_j\}_{j=0,1,\ldots,4}} \frac{1}{N}\sum_{i=1}^{N} u\left(W_5^{(i)}\right)$$

s.t. $\{x_j\}_{j=0,1,\ldots,4}$ are optimal portfolio strategies (6)

where $W_5^{(i)} = B_0 + \sum_{j=0}^{4} x_j p_j^{(i)} B_j$ is the $i-th$ observation of the final wealth and $p_t^{(i)} = [p_{1,t}^{(i)}, \ldots, p_{n,t}^{(i)}]'$ is the $i-th$ observation of the vector of excess returns $p_{k,t}^{(i)} = r_{k,t}^{(i)} - r_{0,t}$ relative to the $t-th$ period.

Table 2. Maximum expected utility $\max_{\{x_j\}_{j=0,1,\ldots,T-1}} E\left(1\text{-}\exp\left(-\frac{1}{a}W_T\right)\right)$ for $a = 0.5,1,1.5,2$ under three different distributional hypotheses: BM, VG, and NIG and three different term structures

	TERM1			TERM2			TERM3		
	BM	VG	NIG	BM	VG	NIG	BM	VG	NIG
a=0.5	0.8727	**0.8731**	0.8728	0.8727	**0.8731**	0.8728	0.8726	**0.8732**	0.8730
a=1	0.6468	**0.6479**	0.6473	0.6471	**0.6485**	0.6479	0.6477	**0.6491**	0.6484
a=1.5	0.5037	**0.5053**	0.5045	0.5044	**0.5062**	0.5053	0.5052	**0.5073**	0.5063
a=2	0.4117	**0.4136**	0.4126	0.4127	**0.4148**	0.4137	0.4138	**0.4161**	0.4150

Table 3. Maximum expected utility $\max_{\{x_j\}_{j=0,1,\ldots,T-1}} E\left(u\left(W_T\right)\right)$, where u(x) is given by (8) for $c = 1,2,3,4,5$ under three different distributional hypotheses: BM, VG, and NIG and three different term structures

	TERM1			TERM2			TERM3		
	BM	VG	NIG	BM	VG	NIG	BM	VG	NIG
c=1	0.9942	**0.9973**	0.9964	0.9983	**1.0025**	1.0012	1.0031	**1.0083**	1.0065
c=2	1.5396	**1.5422**	1.5407	1.5404	**1.5436**	1.5422	1.5419	**1.5453**	1.5436
c=3	2.8763	**2.8994**	2.8880	2.8910	**2.9168**	2.9043	2.9073	**2.9361**	2.9225
c=4	4.3106	**4.3799**	4.3454	4.3578	**4.4359**	4.3980	4.4106	**4.4974**	4.4565
c=5	5.9522	**6.1025**	6.0276	6.0572	**6.2264**	6.1443	6.1745	**6.3623**	6.2738

Finally, we obtain Tables 2 and 3 with the approximated maximum expected utility considering the three implicit term structures. In fact, we implicitly assume the approximation $\frac{1}{N}\sum_{i=1}^{N} u\left(W_5^{(i)}\right) \approx E\left(u\left(W_5^{(i)}\right)\right)$. Tables 2 and 3 show a superior performance of Lévy non Gaussian models with respect to the Gaussian one by the point of view of investors that maximize expected utility (7) and (9). In particular, the Variance Gamma model presents the best performance for different utility functions and term structures. Thus, from an ex-ante comparison among Variance Gamma, Normal

Inverse Gaussian and Brownian motion models, investors characterized by the utility functions (7) and (8) should select portfolios assuming a Variance Gamma distribution.

Table 4. Ex-post final wealth obtained by the optimal strategies solutions of the problem max $E(1-\exp(-W_T/a))$ for a=0.5, a=1, a=1.5, a=2 under three different distributional hypotheses: BM, VG, and NIG and three different term structures

	TERM1			TERM2			TERM3		
	BM	VG	NIG	BM	VG	NIG	BM	VG	NIG
a=0.5	1.0762	**1.0805**	1.0711	1.0716	**1.0755**	1.0670	1.0672	**1.0966**	1.0851
a=1	1.1319	**1.1404**	1.1470	1.1498	**1.1594**	1.1381	1.1401	**1.1487**	1.1295
a=1.5	1.1876	**1.2304**	1.1976	1.2019	**1.2154**	1.1856	1.1888	**1.2269**	1.1961
a=2	1.2433	**1.2903**	1.2482	1.2540	**1.2993**	1.2568	1.2617	**1.3050**	1.2627

Table 5. Ex-post final wealth obtained by the optimal strategies solutions of the problem max $E(u(W_T))$ where u(x) is given by (8) for $c = 1,2,3,4,5$ under three different distributional hypotheses: BM, VG, and NIG and three different term structures

	TERM1			TERM2			TERM3		
	BM	VG	NIG	BM	VG	NIG	BM	VG	NIG
c=1	1.3269	**1.3503**	1.3241	1.3322	**1.3553**	1.3279	1.3347	**1.3571**	1.3294
c=2	1.1319	**1.1404**	1.1217	1.1237	**1.1594**	1.1381	1.1401	**1.1487**	1.1295
c=3	1.4383	**1.4702**	1.4252	1.4364	**1.4951**	1.4228	1.4320	**1.4873**	1.4404
c=4	1.8281	**1.9198**	1.8299	1.8532	**1.9428**	1.8261	1.8698	**1.9561**	1.8622
c=5	2.3573	**2.5194**	2.3358	2.3743	**2.5583**	2.3717	2.4049	**2.5812**	2.3951

The term structure determines the biggest differences in the portfolio weights of the same strategy and different periods. When the interest rates of the implicit term structure are growing (decreasing) we obtain that the investors are more (less) attracted to invest in the riskless in the sequent periods. Generally it does not exist a common factor between portfolio weights of different periods and of the same strategy. However, when we consider the flat term structure (2-nd term structure), the portfolio weights change over the time with the same capitalization factor.

In Table 4 and 5 we show the ex-post final wealth under the three term structures for the three distributional assumptions and for the two utility functions. The results confirm the better performance of the Variance Gamma approach with respect to the Normal Inverse Gaussian and Brownian motion ones. Moreover in this ex-post comparison we observe a better performance of the Brownian motion with respect to the NIG model.

4 Concluding Remarks

This paper proposes an empirical comparison among three distributional hypotheses based on Lévy processes. We discuss the portfolio optimization problem by the point of view of investors that maximize either exponential utility functions or quadratic-logarithm utility functions. Therefore, we propose two models that take into account

the heavier behavior of log-return distribution tails. The ex-ante empirical comparison shows a greater performance of two alternative subordinated Lévy processes. Instead, the ex-post comparison, even though it confirms the better behavior of the Variance Gamma model, shows a better performance of the Brownian motion model with respect to the Normal Inverse Gaussian one. However, several further empirical analysis should be necessary to validate the multi-period models here presented based on the mean, the variance of the final wealth maybe considering even its skewness and kurtosis as suggested by the first part of the paper.

Acknowledgments

The authors thank for grants COFIN 60% 2006, 2007 and, for helpful comments, seminar audiences at EURO XXII (Praha, 2007).

References

1. Athayde, G.M., Flôres Jr., R.G.: Finding a maximum skewness portfolio – a general solution to three-moments portfolio choice. Journal of Economic Dynamics and Control 28, 1335–1352 (2004)
2. Cont, R., Tankov, P.: Financial modeling with jump processes. Chapman & Hall, Boca Raton (2004)
3. Ortobelli, L.S.: The classification of parametric choices under uncertainty: analysis of the portfolio choice problem. Theory and Decision 51, 297–327 (2001)
4. Ortobelli, L.S., Rachev, S., Huber, I., Biglova, A.: Optimal portfolio selection and Risk management: A comparison between the stable paretian approach and the Gaussian one. In: Handbook of Computational and Numerical Methods in Finance, Birkhauser, Boston, ch. 6, pp. 197–252 (2004)
5. Rachev, S., Mittnik, S.: Stable Paretian Models in Finance. John Wiley & Sons, New York (2000)
6. Staino, A., Ortobelli, L.S., Massabò, I.: A comparison among portfolio selection strategies with subordinated Lévy processes. International Journal of Compute Science and Network Security 7(7) (to appear, 2007)

Analyzing the Influence of Overconfident Investors on Financial Markets Through Agent-Based Model

Hiroshi Takahashi and Takao Terano

Graduate School of Humanities and Social Sciences, Okayama University, 3-1-1 Tsushima-naka, Okayama-city, 700-8530, Japan
Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, 4259 Nagatsuda-Cho, Midori-ku, Yokohama 226-8502, Japan
htaka@e.okayama-u.ac.jp, terano@dis.titech.ac.jp

Abstract. In this research, we employ Agent-Based Model to analyze how asset prices are affected by investors' Behavior. This analysis places focus on the influence of overconfident investors on financial market. As a result of intensive analysis, we find that overconfident investors are generated in a bottom-up fashion in the market. Furthermore, it has also been found that overconfident investors have the ability to contribute to market efficiency.

Keywords: Agent Based-Model, Asset Pricing, Behavioral Economics, Overconfidence.

1 Introduction

In the area of computer science, Agent-Based Model is proposed as an effective method to analyze the relation between micro-rules and macro-behavior [3] [2]. Agent-Based Model is an attempt to explain the macro-behavior of systems by local rules. As a result of applying this Model to social science, it is found that a variety of macro-behavior emerges bottom-up from local micro-rules [7]. An artificial market is one of the good applications of Agent-Based Model to financial markets [1] [20] [15] [1].Recently various kinds of financial services have been proposed. In order to improve them, it is necessary to analyze financial markets from the bottom up. Agent-Based Model is effective method for them.

In recent years, there has been rising interest in a field called behavioral finance which incorporates psychological methods in analyzing investor behavior. There are numerous arguments in behavioral finance that investors' decision making bias can explain phenomenon in the financial market which until now had gone unexplained by pointing out limit to arbitrage and existence of systematic biases in decision Making [17] [13] [14].

However, there is also criticism that most such arguments in behavioral finance are simply ad hoc, applying decision making bias exogenously, and only

[1] Hirshleifer describes how Agent-Based Model is effective to analyze financial markets [12].

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1042–1052, 2007.

introducing decision making bias conveniently in order to explain certain phenomenon in the financial market.

With such underlying arguments, this analysis aims to show that decision making bias discussed in financial economics appears in a bottom-up fashion in the financial market. Above all, this research is undertaken with a focus on overconfident decision making which has been under the spotlight in recent years[4] [10] [18]. Furthermore, this research inquires into the conditions under which transaction prices reflect fundamental values.

The next section of this paper explains the model utilized for this analysis-Cbefore analysis results are looked at in section 3. Section 4 contains a summary.

2 Model

A computer simulation of the financial market involving 1000 investors was used as the model for this research, shares and risk-free assets being the 2 possible transaction methods [1] [19]. Several types of investors exist in the market, each undertaking transactions based on their own stock valuations. This market is composed of 3 major steps, (1) generation of corporate earnings, (2) formation of investor forecasts, (3) setting transaction prices. The market advances through repetition of these steps[19].

2.1 Negotiable Assets in the Market

This market consists of both risk-free and risky assets. There is a financial security (as risky assets) in which all profits gained during each term are distributed to the shareholders. The corporate earning (y_t) accrues according to the process of $y_t = y_{t-1} + \varepsilon_t, where\, \varepsilon_t \sim N(0, \sigma_y^2)$[16], and the stock is traded after the corporate profit of current period is announced. Each investor is given common asset holdings at the start of the term and is able to borrow or lend the risk-free asset unlimitedly in principle. The initial asset amount of every investor is 1,000 in stock and 1,000 in risk-free asset.

2.2 Modeling Investor Behavior

Investors in the market evaluate transaction prices based on their own forecast for market tendency, taking into account both risk and return rates when making investment decisions. Each investor decides on the investment ratio (w_t) of stock for each term based on the maximum objective function of $f(w_t^i) = r_{t+1}^{int,i} \cdot w_t^i + r_f \cdot (1 - w_t^i) - \lambda \cdot (\sigma_{t-1}^{s,i})^2 (w_t^i)^2$. In this case, $r_{t+1}^{int,i}$ and $\sigma_{t-1}^{s,i}$ express the expected rate of return and risk for stock as estimated by each investor i. r_f represents the risk-free rate. w_t^i is the stock investment ratio of investor i for term t [6] [19].

Expected rate of return for shares ($r_{t+1}^{int,i}$) is calculated as $r_{t+1}^{int,i} = (1 \cdot c^{-1} \cdot (\sigma_{t-1}^{s,i})^{-2}) / (1 \cdot c^{-1} \cdot (\sigma_{t-1}^{s,i})^{-2} + 1 \cdot (\sigma_{t-1}^{s,i})^{-2}) \cdot r_{t+1}^{f,i} + (1 \cdot (\sigma_{t-1}^{s,i})^{-2}) / (1 \cdot c^{-1} \cdot (\sigma_{t-1}^{s,i})^{-2} + 1 \cdot (\sigma_{t-1}^{s,i})^{-2}) \cdot r_t^{im}$. Here, $r_{t+1}^{f,i}$,r_t^{im} express the expected rate of return, calculated

respectively from short-term expected rate of return, and risk and gross current price ratio of stock etc[6] [19]. c is adjustment coefficient[6][2].

Short-term expected rate of return ($r_t^{f,i}$) is obtained by $r_{t+1}^{f,i} = ((P_{t+1}^{f,i} + y_{t+1}^{f,i})/P_t - 1) \cdot (1 + \eta_t^i)$, $(P_{t+1}^{f,i}, y_{t+1}^{f,i})$ being the equity price and profit forecast for term t+1 as estimated by the investor. Short-term expected rate of return includes the error term ($\eta_t^i \sim N(0, \sigma_n^2)$) reflecting that even investors of the same forecast model vary slightly in their detailed outlook.

Expected rate of return for stock (r_t^{im}) as obtained from stock risk etc. is calculated from stock risk($\sigma_{t-1}^{s,i}$), benchmark equity stake(W_{t-1}), investors' degree of risk avoidance(λ), and risk-free rate(r_f) in the equation $r_t^{im} = 2\lambda(\sigma_{t-1}^s)^2 W_{t-1} + r_f$.

2.3 Equity Price Forecasting Method

In this research, we analyze several kinds of forecasting methods such as (1) forecasting based on fundamental values, (2) forecasting based on trends (4 types), and (3) forecasting based on past averages (4 types).

The fundamental value of shares is estimated using the dividend discount model. Fundamentalists estimate the forecast stock price ($P_{t+1}^{f,i}$) and forecast profit ($y_{t+1}^{f,i}$) from profit for the term (y_t) and discount rate of stock (δ) respectively as $P_{t+1}^{f,i} = y_t/\delta$,$y_{t+1}^{f,i} = y_t$.

Forecasting based on trends involves forecasting next term equity prices and profit through extrapolation of the most recent stock value fluctuation trends. In this research we deal with 4 types of trend measurement period: 1 day, 5 days, 10 days, and 20 days for trend measurements.

Forecasting based on past averages involves estimating next term equity prices and profit based on the most recent average stock value.In this research we deal with 4 types of average measurement period: 1 day, 5 days, 10 days, and 20 days.

2.4 Risk Estimation Method

Stock risk is measured as $\sigma_{t-1}^{s,i} = s_i \cdot \sigma_{t-1}^h$. In this case, σ_{t-1}^h is an index that represents stock volatility calculated from price fluctuation of the most recent 100 steps, and s_i is the degree of overconfidence. The presence of a strong degree of overconfidence can be concluded when the value of s_i is less than 1, as estimated forecast error is shown as lower than its actual value. The investors whoes value of s_i is less than 1 tend to invest more actively[3].

2.5 Deciding Transaction Prices

Transaction prices are set as the price where stock supply and demand converge[1]. The investment ratio (w_t^i) is the decreasing function of the stock price, and the total number of the stock issued in the market (N) is constant. We derive the traded price where the demand meets the supply($\sum_{i=1}^{M}(F_t^i w_t^i)/P_t = N$).

[2] For more detail, refer to Black/Litterman[6].

[3] e.g. When such investors predict that stock price will increase, they invest more in stock than ones whoes value of s_i is 1.

2.6 Rules of Natural Selection

After 25 terms pass since the market has started, the rules of natural selection come into play in this market on the basis of cumulative excess return for the most recent 5 terms [11] [19]. The rules of natural selection are composed of the 2 steps of (1) appointment of investors who alter their investment strategy(forecast type and degree of overconfident(s_i)), and (2) alteration of investment strategy.

At first step, the investor who obtain negative cumulative excess return changes the strategy at the following probability:$p_i = min(1, max(0.5 \cdot e^{-r_i^{cum}} - 0.5, 0))$, where p_i is probability at which investor i changes own strategy and r_i^{cum} is cumulative return of investor i during recent 5 terms.

At second step, the investors who change the strategy tend to select the strategy that has brought positive cumulative excess return. The probability to select $strategy_i$ as new strategy is given as:$p_i = \frac{e^{(r_i^{cum})}}{\sum_{j=1}^{M} e^{(r_j^{cum})}}$, where r_i^{cum} is the cumulative excess return of each investor.

3 Method of Analysis

This analysis sets out to search for conditions by which the market value would reflect the fundamental value, after firstly undertaking a conditional search for investment strategy capable of acquiring excess return.

3.1 Searching for Investment Strategy

Firstly, analysis was made of the initial forecasting model ratio where there was a (1) high ratio of fundamental forecasting, and (2) a high ratio of trend forecasting. As the results of this analysis confirmed a strengthening degree of overconfidence in both cases, an analysis of (3) the random distribution of the initial ratio of each forecasting model was also undertaken to determine whether the same result could be obtained under different conditions. The results of this analysis are explained in detail below.

When there is a High Ratio of Fundamental Forecasting. As fundamentalists enforce a strong influence on the market value under these conditions, the market value is almost in concord with the fundamental value (Fig. 1).@It can be confirmed that the number of fundamentalists is on the increase due to the rules of natural selection in regard to the transition of investor numbers (Fig.2). Looking at transition in the degree of overconfidence, a strengthening degree of overconfidence can be confirmed in the remaining investors as market transactions go forward(Fig. 3).

When there is a High Ratio of Trend Forecasting. When there is a high ratio of investors using trend forecasting, the market value deviated greatly from the fundamental value. It was confirmed that the number of investors using trend forecasting also increases as such investors enforce a strong influence on

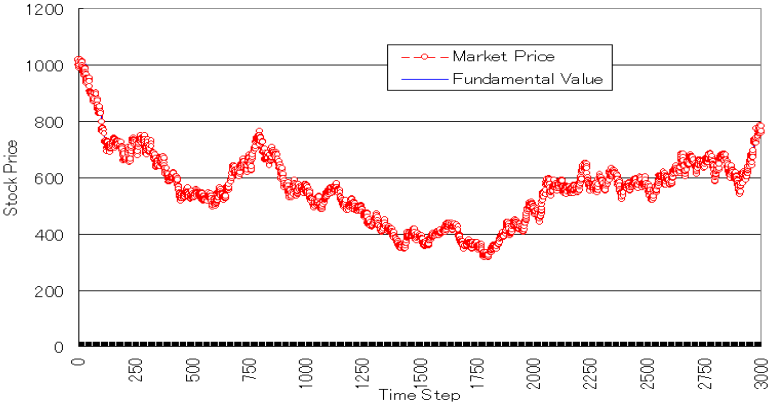


Fig. 1. Price history(Fundamentalist:Trend=500:500)

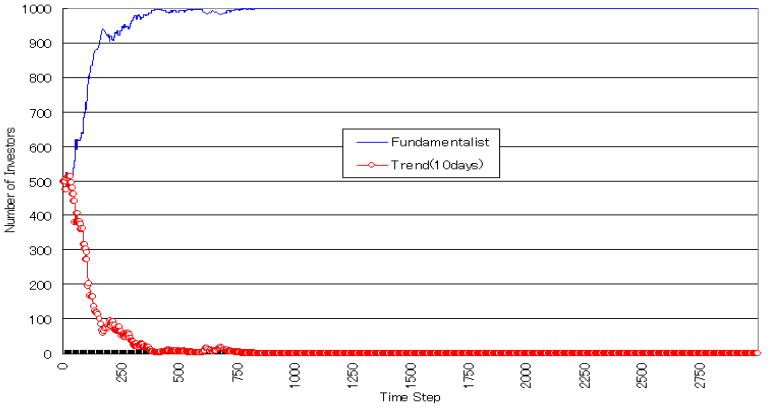


Fig. 2. History of the number of Investors(Fundamentalist:Trend=500:500)

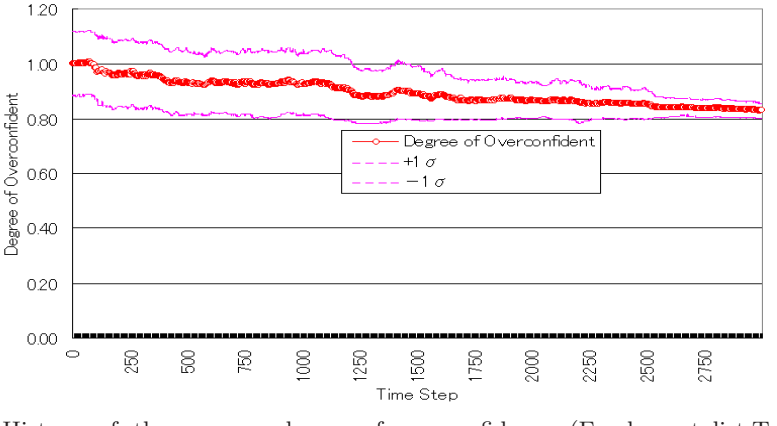


Fig. 3. History of the average degree of overconfidence (Fundamentalist:Trend= 500:500)

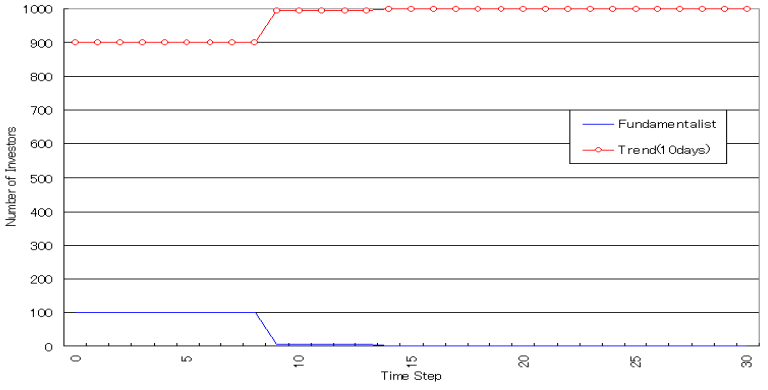


Fig. 4. History of the number of Investors(Fundamentalist:Trend=100:900)

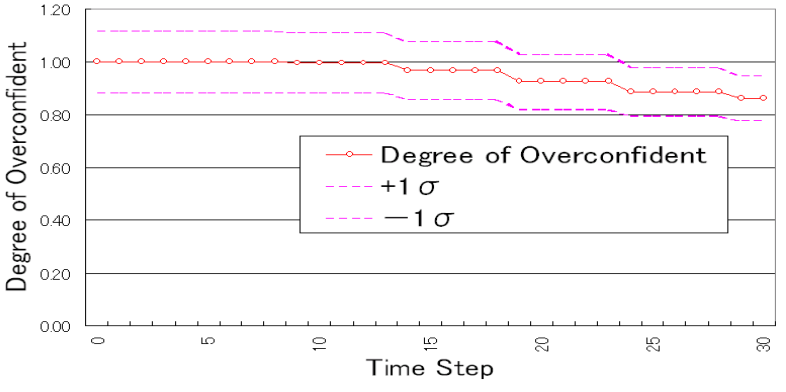


Fig. 5. History of the average degree of overconfidence (Fundamentalist:Trend= 100:900)

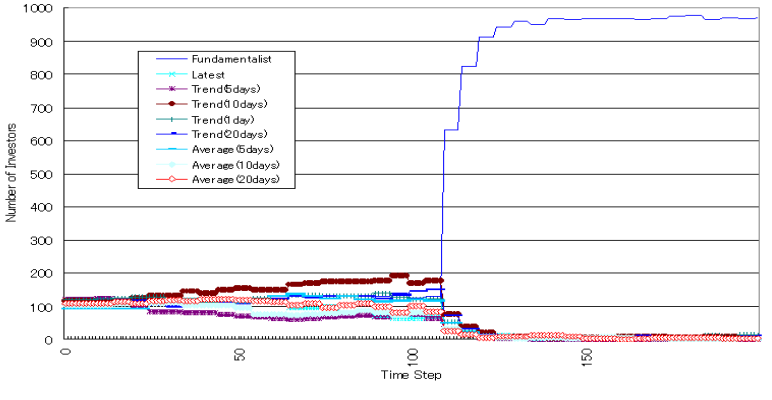


Fig. 6. History of the number of Investors(Random)

the market value(Fig.4). This is thought to be the result of an investment environment in which different forecasting methods were applied to obtain excess return. On the other hand, it was confirmed that investors with a strong degree of overconfidence survive in the market even under these conditions(Fig.5).

When the Initial Ratio is Applied Randomly. A case in which the initial ratio of investors is applied randomly was analyzed next. Although the case example (Fig. 4) shown here indicates that numerous investors employ investment strategy based on the fundamental value, the forecasting model employed by market investors is dependant on the ratio of each type of investor etc, changing along with circumstances such as trend forecasting and the average value of past equity prices. In contrast, it has been confirmed that overconfident investors survive in the market even when a random initial value is applied for the degree of overconfidence (Fig. 5).

In this analysis, we are also able to confirm that overconfident investors emerges from the bottom up. This interesting analysis result suggests the possibility of universality when survival trends of overconfident investors are compared with the forecasting model[4].

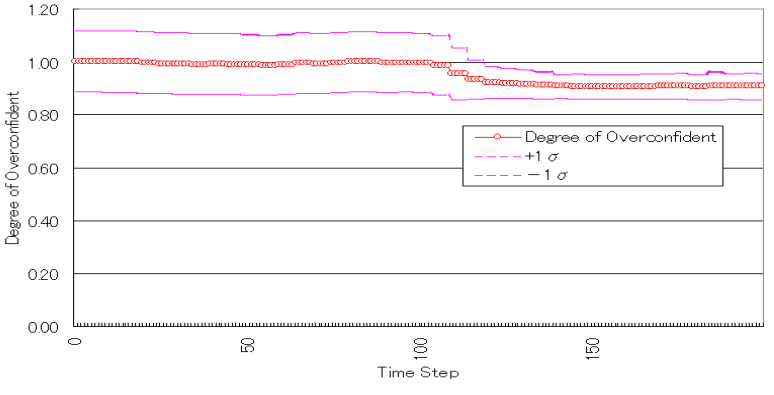


Fig. 7. History of the degree of overconfidence(Random)

3.2 Exploring Market Conditions

This analysis endeavors to determine the conditions necessary for transaction prices to reach the fundamental value.

Inverse Simulation Analysis Method. Inverse Simulation Analysis consists of the following 3 steps. (1) Carry out 100 times a simulation with an investment period of 100 terms. (2) Calculate the index of deviation between transaction prices and the fundamental value for each simulation. (3) Set the calculated index

[4] In this mean, overconfidence in this paper is modeled neither ad-hoc nor conveniently.

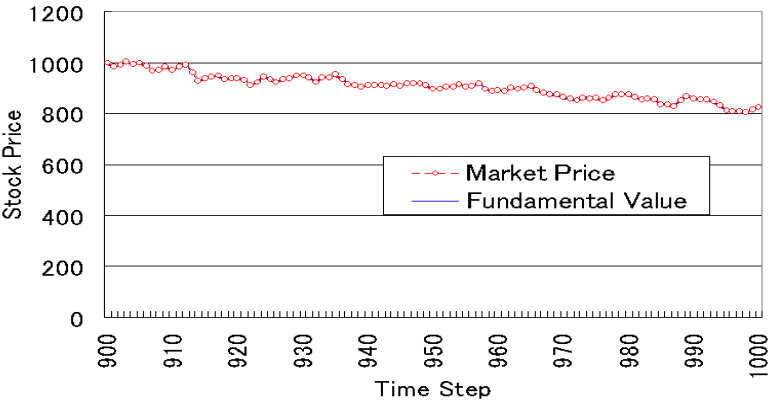


Fig. 8. Price History(Inverse Simulation)

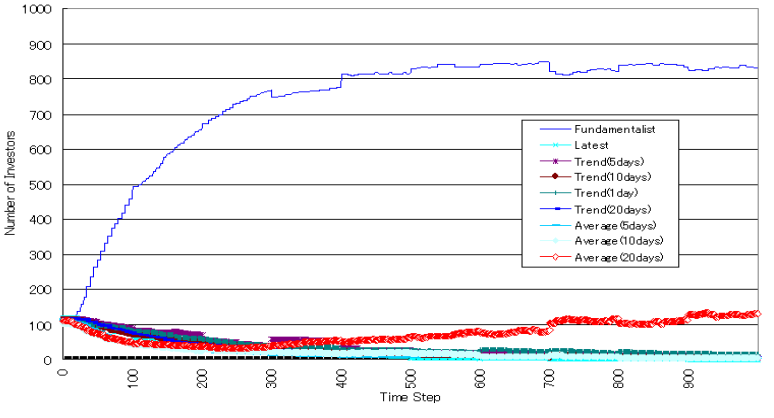


Fig. 9. History of the average number of Investors(Inverse Simulation)

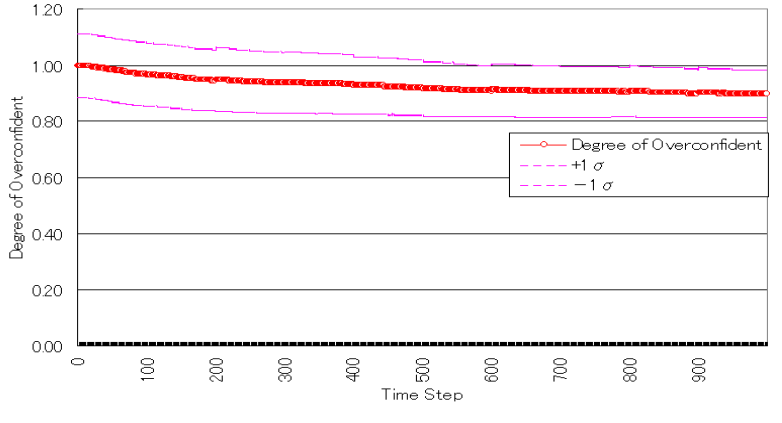


Fig. 10. History of the average degree of overconfidence(Inverse Simulation)

as the adaptive value and select 100 simulation conditions (investors' forecasts, confidence). This analysis is undertaken through repetition of these 3 steps. The index (q) of deviation between transaction prices and the fundamental value expresses the deviation ratio with the fundamental value and is specifically calculated as $q = E[x]^2 + Var[x]$, where P_t^0 is the fundamental value for term t and $x_t = (P_t - P_t^0)/P_t^0$.

3.3 Conditional Search Results

It can be seen from analysis results that transaction prices tend to approach the fundamental value (Fig. 8) when there is a high percentage of fundamentalist investors (Fig. 9) coupled with a strong degree of investor confidence (Fig. 10). In addition, transaction prices almost match the fundamental value in this case.

Traditional finance argues that investors who are able to swiftly and accurately estimate both the risk and rate of return on stock survive in the market, and such investment behaviors contribute to market efficiency[5]. However, analysis results obtained here regarding the influence irrational investors have on prices suggests a different situation, pointing to the difficulty of market modeling which takes real conditions into account. These results indicate that overconfident investors have pricing power and they can contribute to efficient market when assumptions of traditional financial theory are extended to the ones closer to the reality.

4 Summary

This paper utilizes the Agent-Based Model to analyze both microscopic and macroscopic associations in the financial market. In the process, it has been found that overconfident investors are generated in a bottom-up fashion in the market. Showing the existence of a survival mechanism as a characteristic feature of overconfidence in decision making is one of the significant achievements of this research. Furthermore, this research has also succeeded in showing that such characteristic features have the ability to contribute to a market which reflects fundamentals. Future issues include market modeling which takes more realistic conditions into account.

References

1. Arthur, W.B., Holland, J.H., Lebaron, B., Palmer, R.G., Taylor, P.: Asset Pricing under Endogenous Expectations in an Artificial Stock Market. In: The Economoy as an Evolving Complex System II, pp. 15–44. Addison-Wesley, Reading (1997)
2. Axelrod, R.: The Complexity of Cooperation -Agent-Based Model of Competition and Collaboration. Princeton Uniersity Press, Princeton (1997)

[5] Efficiency of the market is one of the most important hypothesis in Financial Economics[8]. In efficienct markets, asset prices swiftly and accurately reflect fundamental values.

3. Axtell, R.: Why Agents? On the Varied Motivation For Agent Computing In the Social Sciences, the Brookings Institution Center on Social and Economic Dynamics Working Paper, vol. 17 (November 2000)
4. Barber, B., Odean, T.: Trading is Hazardous to Your Wealth: The Common Stock Investment Performance of Individual Investors. Journal of Finance 55, 773–806 (2000)
5. Bazerman, M.: Judgment in Managerial Decision Making. John Wiley & Sons, Chichester (1998)
6. Black, F., Litterman, R.: Global Portfolio Optimization. Financial Analysts Journal, 28–43 (September-October 1992)
7. Epstein, J.M., Axtell, R.: Growing Artificial Societies Social Science From The Bottom Up. MIT Press, Cambridge (1996)
8. Fama, E.: Efficient Capital Markets: A Review of Theory and Empirical Work. Journal of Finance 25, 383–417 (1970)
9. Fuller, R., Farrel Jr., J.: Modern Investment and Security Analysis. McGraw-Hill, New York (1987)
10. Gervais, S., Odean, T.: Learning to be overconfident. Review of Financial Studies 14(1), 1–27 (2001)
11. Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
12. Hirshleifer, D.: Investor Psychology and Asset Pricing. Journal of Finance 56, 1533–1597 (2001)
13. Kahneman, D., Tversky, A.: Prospect Theory of Decisions under Risk. Econometrica 47, 263–291 (1979)
14. Kahneman, D., Tversky, A.: Advances in prospect Theory: Cumulative representation of Uncertainty. Journal of Risk and Uncertainty 5 (1992)
15. Levy, M., Levy, H., Salomon, S.: Microscopic Simulation of Financial Markets. Academic Press, London (2000)
16. O'Brien, P.: Analysts' Forecasts as Earnings Expectations. Journal of Accounting and Economics, 53–83 (January 1988)
17. Shleifer, A.: Inefficient Markets. Oxford University Press, Oxford (2000)
18. Stein, J.C.: Agency, Information and Corporate Investment, NBER Working Paper, vol. 8342 (2001)
19. Takahashi, H., Terano, T.: Agent-Based Approach to Investors' Behavior and Asset Price Fluctuation in Financial Markets. Journal of Artificial Societies and Social Simulation 6, 3 (2003)
20. Tesfatsion, L.: Agent-Based Computational Economics. Economics Working Paper, vol. 1, Iowa Sate University (2002)

A Prameter List

List of the principle parameters used in this analysis.
M: Number of Investors (1000)
N: Number of shares (1000)
F_t^i: Total Assets value of investor i for Term t ($F_0^i = 2000$: common)
W_t: Benchmark equity stake for term t ($W_0 = 0.5$)
w_t^i: Equity stake of investor i for term t ($w_0^i = 0.5$: common)
y_t: Profits generated during term t ($y_0 = 0.5$)

σ_y: Standard deviation of profit fluctuation ($0.2/\sqrt{200}$)
δ: Discount rate of shares (0.1/200)
λ: Invstors' degree of risk avoidance (1.25)
r_t^{im}: Expected rate of share return as estimated from risk etc
c: adjustment coefficient (0.01)
σ_t^s: Assessed value of standard deviation of share fluctuation
σ_t^h: Historical stock volatility
P_t: Transaction prices for term t
$P_t^{f(,i)}$: Forecast value of transaction prices (of investor i) for term t
$y_t^{f(,i)}$: Forecast value of profits (of investor i) for term t
$r^{f(,i)}$: Short term expected rate of return on shares (of investor i)
σ_n: Standard deviation of data dispersion for short term expected rate of return on shares (0.01)
a_t: Price trend on stock until term t
r_i^{cum}: Cumulative excess return of investor i for most recent 5 term
p_i: Probability that investors' who alter their strategy will adopt investor i's strategy
s_i: Coefficient to express degree of confidence (uniform random number of 0.8-1.2)
a: Coefficient that expresses the degree of investment strategy selectivity (20)

Modularity, Product Innovation, and Consumer Satisfaction: An Agent-Based Approach

Shu-Heng Chen and Bin-Tzong Chie

AI-ECON Research Center, Department of Economics
National Chengchi University, Taipei, Taiwan 11623
chchen@nccu.edu.tw, g0258501@nccu.edu.tw

Abstract. The importance of modularity in product innovation is analyzed in this paper. Through simulations with an agent-based modular economic model, we examine the significance of the use of a modular structure in new product designs in terms of its impacts upon customer satisfaction and firms' competitiveness. To achieve the above purpose, the automatically defined terminal is proposed and is used to modify the simple genetic programming.

1 Motivation and Introduction

This work is a continuation of [1,2], which provide an agent-based model that simulates the evolution of product innovation by growing it from the bottom up. The earlier work is not just to provide an agent-based approach, but to introduce a new representation of commodities, production, and preference, via the use of *genetic programming* (GP). However, [1,2] only consider simple genetic programming [3]. The end result is that in many of their early simulations, only primitive desires are satisfied, and the economy can rarely advance to a mature state where consumers' desires can be met a sophisticated degree. One cause of this problem is that simple GP is not an appropriate tool to work with the idea of *functional modularity* (to be detailed in Sect. 3). This limitation has been long realized by GP researchers, e.g., [4]. In this paper, we remedy this problem by replacing the simple GP with automatically defined terminals (ADTs), which are very similar in spirit to automatically defined functions (ADFs), invented by John Koza [4]. As Koza pointed out, devices of this kind can provide some hierarchical mechanism to exploit modularities inherent in problem environments.

With this modified version of GP, two experiments are carried out. The first experiment examines the contribution of functional modularity to consumers' satisfaction. The second series of experiments then examines the importance of modularity in competition among firms. We simulate an agent-based economy to allow the firm that designs new products using a modular structure to compete with the firm that does not. In a sense, this is equivalent to replicating the well-known story regarding the competition between Hora and Tempus, two

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1053–1062, 2007.

imaginary watchmakers offered by Herbert Simon in his celebrated work on *the architecture of complexity* [5].

The rest of the paper is organized as follows. Sect. 2 provides a brief review of the agent-based modular economy introduced in [1,2]. Sect. 3 proposes the automatically defined terminal and motivates this idea with its connection to hierarchical modularity. The two experiments are conducted in Sect. 4 and 5. Sect. 6 gives the concluding remarks.

2 The Agent-Based Modular Economy

The economic model on which our simulation of product innovation is based is largely the same as [1,2], which we shall briefly review here. Chen and Chie [1,2] considered an economy composed of a number of firms and consumers with the usual objectives. To maximize profits, firms have the incentive to search for the products which can satisfy consumers to the highest degree. In the meantime, consumers allocate their limited budget to commodities which provide them with the greatest degree of enjoyment (measured in terms of consumer's surplus). The interaction between consumers and producers drives the evolution of a series of new products (the innovation process), as shown in Fig. 1.

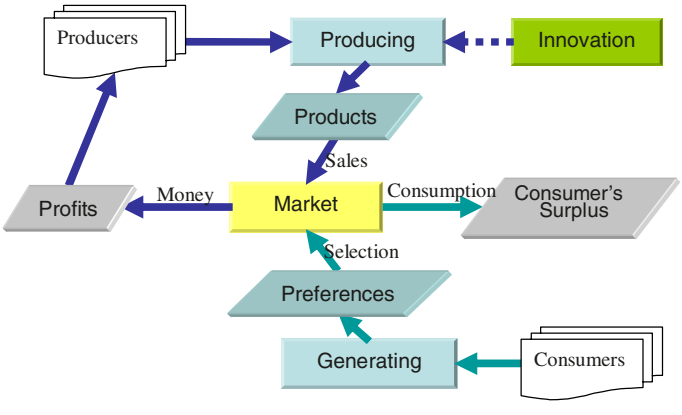


Fig. 1. The agent-based modular economy

The commodity in this economy is represented as a *parse tree* as shown in the first row of Fig. 2. Each parse tree corresponds to a LISP program. The very bottom of the tree, i.e., the leaves, corresponds to the raw inputs (materials) $X1$, $X2$,..., whereas the root and all intermediate nodes represent the processors, $F1$, $F2$,..., applied to these raw materials in a bottom-up order, as the usual behavior of a LISP program. The whole parse tree can, therefore, be interpreted as a production process associated with the commodity. The unit cost of the commodity is a positive function of the number of processors and the number

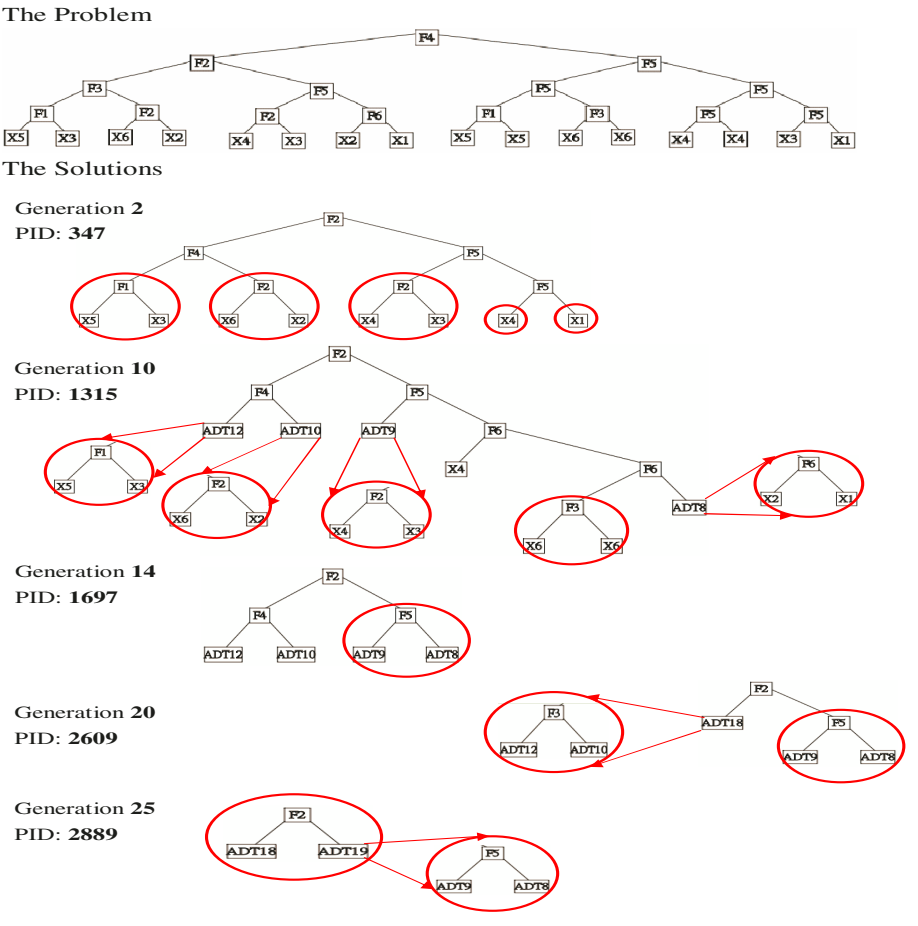


Fig. 2. An illustration of a process of product innovation

of raw inputs, i.e., it is a positive function of the (node) complexity of the commodity. In a simple way, we assume that the unit cost is a linear function of the node complexity.

In each market period, the firm has to decide how to allocate her limited working capital (budget) to R&D (designs of new products), to the production of existing commodities with different quantities and to reserves. R&D is the sole source of new products and is implemented by genetic programming with or without automatically defined terminals (to be detailed in Sect. 3), as shown in Fig. 3.

The preference of the consumers in the economy is also represented by a parse tree. To make the preference tree economically meaningful, three assumptions have been made [2], namely, the monotone, synergy, and consistency condition. The utility from consuming a commodity is based on the module-matching

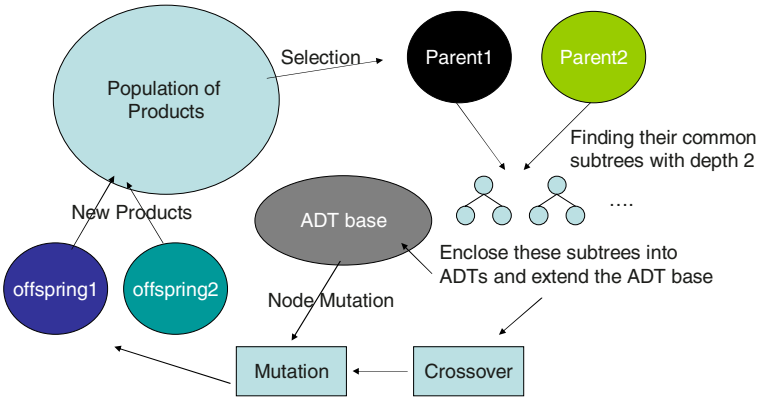


Fig. 3. Genetic programming as an engine for R&D

algorithm proposed in [2]. The idea is to match each possible module (subtree) of the commodity with each possible module of the preference in descending order in relation to the depth of the tree. So, the big modules are matched first; if one succeeds, we sop, and if one fails, we move o the next biggest one. To satisfy the synergy condition and hence the idea of added-value, [2] assumes a power utility function for the preference tree as shown in Fig. 4. As a result, the utility is increasing at a faster rate when higher and higher levels of modular preferences are satisfied.

3 Modularity and Automatically Defined Terminals

Simple genetic programming is designed to find the solution to a problem, which is represented by a *parse tree*. In our case, a solution is analogous to a product, and whose corresponding problem is the ideal product which can bring the highest enjoyment to a target consumer (Fig. 2). The parse tree, from the bottom to the top, can be read as how the solution (product) can be constructed in *parallel*, as well as *incrementally* and *progressively*. What is accomplished at each incremental and parallel step is a minimum or marginal effort to combine what has been established in the previous steps.

As an illustration, Fig. 2 traces an artificial history of product innovation. Consider a target consumer whose preference is depicted in the first row of Fig. 2, which can be regarded as the solution to the problem. Firms do not know this design, and have to figure out the best design by themselves. The five products listed below are the designs discovered in generations 2, 10, 14, 20, and 25. These products match the consumer's needs to a higher and higher level. For example, the product PID 2889, i.e., the 2,889th new product designed by the firm, has completely answered the target's need to the entire first half at level four. Nonetheless, this product does not come out all of a sudden; all it has

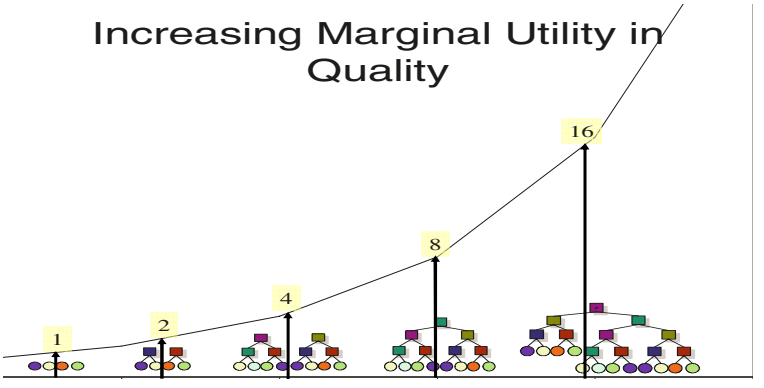


Fig. 4. The power utility function of a preference tree

done is to combine two commodities which were already known before, namely, commodities ADT 18 and ADT 19, both of which were already known to the firm before generation 25. The "marginal" effort here is to assemble them in a right way, i.e., using processor F2.[1]

The results obtained in each step then become the bases or the *building blocks* for the subsequent steps. For example, as shown in Fig. 2, ADT 18 and ADT 19 serve as building blocks for all designs after generation 20. The entire process can then be viewed as a growing but convergent process from leaves to small trees, then to bigger and bigger trees, and finally to the target tree (the solution).

The description above enables us to see how genetic programming can be related to the Simonian notion of complexity [5], i.e., *hierarchy*. Herbert Simon viewed hierarchy as a general principle of complex structures. Hierarchy, he argued, emerges almost inevitably through a wide variety of evolutionary processes, for the simple reason that hierarchical structures are *stable*. To demonstrate the importance of a *hierarchical structure* or *modular structure* in production, Simon offered his well-known story about a competition between Hora and Tempus, two imaginary watchmakers. In this story, Hora prospered because he used the modular structure in his design of watches, whereas Tempus failed to prosper because his design was not modular. Therefore, the story is mainly about a lesson: the advantage of using a modular structure in production.

While using parse tree as the representation, simple genetic programming is not good at using a modular structure. The standard crossover and mutation can easily destroy the already established structure, which may cause the whole discovery or learning process to be non-incremental and non-progressive, and hence very inefficient. This problem is well-known in the GP literature, and has

[1] Of course, from an *ex ante* view, knowing what to combine and in which way is not trivial. In fact, in this example, it took the firm five generations to learn this. In this sense, the contribution is not entirely *marginal*. However, from an *ex post* view, it is just a combination of what we already known.

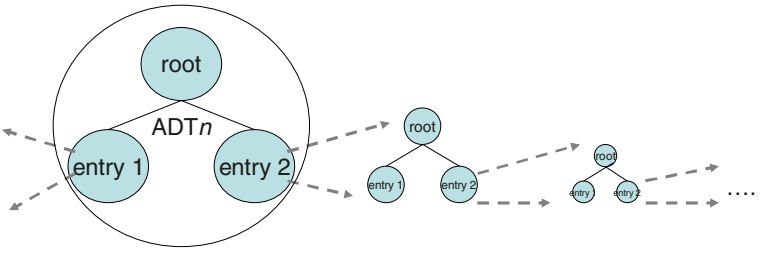


Fig. 5. Automatically defined terminals

been extensively studied with various treatments [6,7,4,8]. Motivated by these earlier studies, we propose automatically defined terminals (ADTs) as a way to enhance GP to find structured solutions.

An ADT, as shown in Fig. 5, is very similar to the automatically defined function (ADF) [4]. It itself has a fixed structure, in this case, a tree with a depth of two. The root of an ADT can be any function from the primitives (function set), while its leaf can be either a terminal from the primitives (terminal set) or can be any existing ADTs. In this way, it shares the same spirit as an ADT, namely, simplification, reuse, and encapsulation. The last item is particularly important because it means that whatever is inside an ADT will not be further interrupted by crossover and mutation. In this way, ADTs can be considered to be the part of learning in which we have great confidence, and which leaves no room for doubt. Through ADTs we distinguish what is considered to be *knowledge* from what is still in a trial-and-error process. Only the former can then be taken as the building blocks (modules), but not the latter. Without ADTs or equivalents, Simple genetic programming is essentially not designed to develop building blocks; therefore, it is not very good at finding the modular structure inherent in the problem.

4 Modularity and Consumer Satisfaction

Simple genetic programming can also detect a modular structure, but it does so only by chance, and hence it may be very difficult to detect complex modules.[2] To see this, in this section, we simulate how well consumers are served when the firm designs new products with modular GP (standard GP plus ADTs), and then compare the result with that of standard GP.

In this simulation, there are 100 consumers in the market. Each consumer has a preference tree with a depth of six. Viewed from the topmost level (the root level), the preference tree is composed of two modules. The one on the left, having a depth of five as shown in the first row of Fig. 2, is identical among all

[2] To define and measure complexity, Simon [5] advocated the use of a hierarchical measure – the number of successive levels of hierarchical structuring in a system or, in our case, the depth of the parse tree.

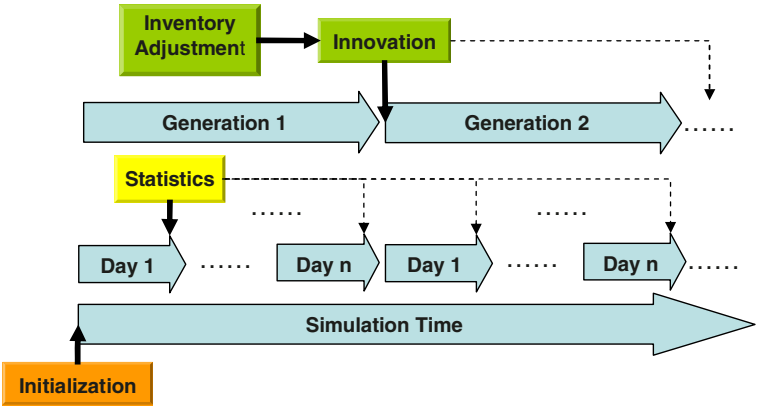


Fig. 6. Market days and learning cycles

consumers, whereas the one on the right, having a depth of five or less, is heterogeneous, and is randomly generated by the ramped half-and-half method, an initialization method frequently used in GP. In this way, consumers' preferences have a common part as well as an idiosyncratic part. For the idiosyncratic part, the complexity is also different.

A profit-maximizing monopoly firm will try to serve the needs of this group of consumers by producing different products with different quantities and also with different degrees of specialization or diversification (customization).[3] The firm has to learn the consumers' preferences and hence, through R&D (driven by GP), design better products. The entire market process is summarized in Fig. 6. The learning cycle (GP cycle) is run with a number of generations (in our case, 5,000). Each generation is composed of a number of trading days (in our case, five). After each learning cycle, the firm has to decide what to produce, including some new products developed via production innovation, how many to produce, and how much to charge. The decision regarding production and R&D is based on the sales and profits statistics collected on the previous market days. The firm then supplies what has been produced, including those new items, during the next few market days.

For further analysis, in each generation, statistics regarding consumer satisfaction are reported. Consumer satisfaction is measured by the actual utility the consumer received from consumption divided by the maximum potential utility the consumer can possibly gain given his preference. The ratio is then multiplied by 1,000, and the measure lies in [0, 1,000]. By averaging the consumer satisfaction over all 100 consumers, we then derive the aggregate consumer satisfaction, which also lies in the same interval. The result is shown in Fig. 7. What Fig. 7 shows is not the result based on a single run, but on fifty runs. Accordingly, what is shown in the left panel of Fig. 7 is the average of the 50 runs, whereas what is shown in the right panel is the maximum of the 50 runs. It can be seen

[3] See [2] for details.

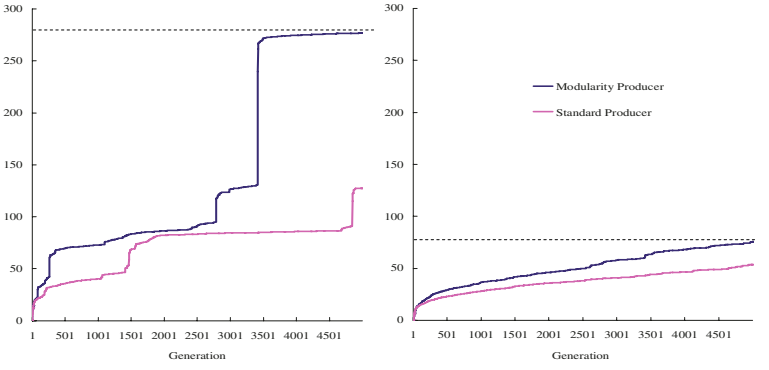


Fig. 7. Modularity and consumer satisfaction

quite easily that the firm whose product design uses a modular structure can satisfy the consumers to a higher degree than the firm whose product design uses a non-modular structure.

5 Modular Structure and Competitiveness

In the previous section, under the assumption of a monopoly firm, we have seen the positive impact of using a modular structure on consumer satisfaction. In this section, we shall pursue this matter further by inquiring about the implications of a modular structure for the competitiveness of firms. In a sense, we attempt to re-examine the story given by Herbert Simon on the competition between two watchmakers: one using a modular structure and one not. For that purpose, we consider a duopolistic competition in which one firm uses a modular structure in her R&D (new product designs) and the other firm does not.

The two duopolistic firms compete with each other in a market composed of 100 consumers whose preferences are partially identical and partially idiosyncratic (see Sect. 4). We then watch their market share, i.e., the total sales of each firm divided by the total sales of the market, and the results are displayed in Fig. 8.[4] The results presented here are not based on a single run, but on one hundred runs. The one shown in the left panel of Fig. 8 is the mean of the 100 runs, whereas the one shown in the right panel is the median of the 100 runs. Below the separation line is the market share owned by the non-modular firm, and above the line is the market share owned by the modular firm. Clearly, their sum equals 100%.

Due to the existence of outliers, the time series behavior of the mean and that of the median is not quite the same, but the eventual dominance of the modular firm is evident. In the first few hundreds of generations, the non-modular firm,

[4] Notice that firms generally produce more than one product and can be very different from each other. Therefore, it is meaningless to measure the market share based on a single product.

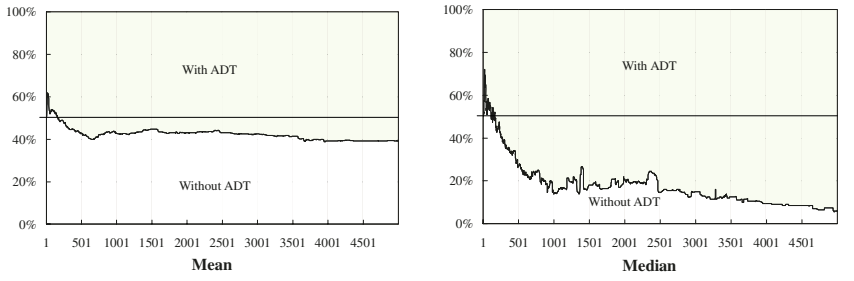


Fig. 8. Modularity and competitiveness

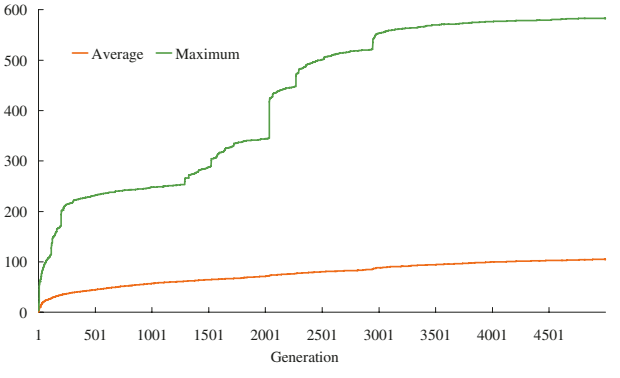


Fig. 9. Consumer satisfaction under competition

however, did have some competitive advantage over the modular firm. This is because establishing modules is costly. The idea of encapsulation associated with ADTs implies a fixed cost and hence a lower degree of mobility, depending on the degree of encapsulation or the complexity of ADTs.[5] Hence, the modular products will generally be more expensive. Unless these products can fit the consumers' needs to a higher degree, these high-priced products will have an adverse influence on marketing. Therefore, there is no guarantee (a probability of one) that the modular firm will always beat the non-modular firm. In fact, in 39 out of our 100 runs, the non-modular firm achieved a higher market share than the modular firm in the last generation (the 5,000th generation).

Finally, as one may expect, competition does bring a better quality to consumers. This is reflected in Fig. 9.

6 Concluding Remarks

Consumers are not random and their behavior can be studied and patterns can be extracted. On the other hand, innovation normally is not a random jump,

[5] See footnote 2 for the measure of complexity.

but follows a gradually changing process. These two together suggest that the economy can be constructed in a modular way, or that the entire economy has a modular structure. In other words, Herbert Simon's notion of the architecture of complexity has the potential to be applied to the whole economy. In this paper, we study the significance of modularity in product innovation. We find that both consumers and producers can benefit from the use of a modular structure in product design. However, modularity may imply a higher cost and less mobility; therefore, its dominance is not certain. Using Simon's story, there is a chance that Tempus prospers and Hora fails.

Acknowledgement. The authors are grateful to Herbert Dawid, Christophre Georges, Massimo Ricottilli, and two anonymous referees for their helpful suggestions. NSC research grant No. 95-2415-H-004-002-MY3 as well as the computer time and facilities of National Center for High-performance Computing are gratefully acknowledged.

References

1. Chen, S.-H., Chie, B.-T.: Agent-Based Economic Modeling of the Evolution of Technology: The Relevance of Functional Modularity and Genetic Programming. International Journal of Modern Physics B 18, 17–19, 2376–2386 (2004)
2. Chen, S.-H., Chie, B.-T.: A Functional Modularity Approach to Agent-based Modeling of the Evolution of Technology. In: Namatame, A., Kaizouji, T., Aruka, Y. (eds.) The Complex Networks of Economic Interactions: Essays in Agent-Based Economics and Econophysics. Lecture Notes in Economics and Mathematical Systems, vol. 567, pp. 165–178. Springer, Heidelberg (2005)
3. Koza, J.R.: Genetic Programming: On the Programming of Computers by the Means of Natural Selection. The MIT Press, Cambridge (1992)
4. Koza, J.R.: Genetic Programming II–Automatic Discovery of Reusable Programs. MIT Press, Cambridge (1994)
5. Simon, H.A.: The Architecture of Complexity. General systems 10, 63–76 (1965)
6. Angeline, P., Pollack, J.: Evolutionary Module Acquisition. In: Proceedings of the 2nd Annual Conference on Evolutionary Programming, pp. 154–163. MIT Press, Cambridge (1993)
7. Hoang, T.-H., Daryl, E., McKay, R., Nguyen, X.H.: Developmental Evaluation in Genetic Programming: The TAG-Based Framework. forthcoming in Knowledge Engineering Review (2007)
8. Rosca, J., Ballard, D.: Hierarchical Self-Organizaion in Genetic Programming. In: Rouveirol, C., Sebag, M. (eds.) Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann, San Francisco (1994)

An Agent-Based Model of Interactions in the Payment Card Market*

Biliana Alexandrova-Kabadjova[1], Andreas Krause[2], and Edward Tsang[1,3]

[1] Centre for Computational Finance and Economic Agents (CCFEA), University of Essex, Wivenhoe Park, Colchester CO4 3SQ, Great Britain
balexa@essex.ac.uk
[2] University of Bath, School of Management, Claverton Down, Bath BA2 7AY, Great Britain
mnsak@bath.ac.uk
[3] Department of Computer Science, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, Great Britain
edward@essex.ac.uk

Abstract. We develop an agent-based model of the competition between payment cards by focusing on the interactions between consumers and merchants determining the subscription and usage of cards. We find that after a short period of time the market will be dominated by a small number of cards, even though there do not exist significant differences between cards and the market is fully competitive. In contrast to the existing literature we focus on the dynamics of market shares and emergence of multi-homing rather than equilibrium outcomes.

Keywords: Two-sided markets, network externalities, agent-based modeling, multi-homing.

1 Introduction

The market for payment cards - more commonly known as credit and debit cards - is dominated by two large competitors, Visa and Mastercard, while the remaining competitors, most notably American Express, Diners Club, Discover and JCB, have a significantly smaller market share. Over the last decade the dominance of the two main competitors has increased with Visa gaining a small advantage over Mastercard through the more widespread use of debit cards issued by Visa. Payment cards are now universally held by consumers in developed countries and accepted by most retailers, making them an important service provider for administering the payments for purchases. Understanding the dynamics of the competition between payment cards is essential for any potential regulation of this market. In this paper we present a novel approach to modeling this market for payment cards by using an agent-based approach focusing on the behavior of card holders and merchants rather than direct competition between payment cards via their fee structure.

* We acknowledge the financial support of the Consejo Nacional de Ciencia y Tecnología (CONACYT).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1063–1072, 2007.

The payment card market is a two-sided market in which for a successful transaction involving a card both participants have to hold the payment card (buyer/consumer) and accept the card as payment (seller/merchant), respectively. The benefits from holding (accepting) a specific card obviously depends on how often the card can be used. The more merchants (consumers) accept (hold) the card, the larger the benefits to the customer (merchant). The existence of these network externalities is an important aspect of the competition between different cards.

The main focus of the literature has for a long time been on the fee structure of payment cards, with emphasis placed on the interchange fee. The interchange fee is the amount the bank of a merchant pays the bank of the consumer for each transaction with a card. The extensive literature in this field can generally be divided into models with only a single payment card, see e.g [1,2,3,4], and those that allow for competition between payment cards, see e.g. [5,6].

More recently a growing amount of literature has addressed the problem of multi-homing in payment card markets, i.e. consumers holding or merchants accepting more than one payment card. In this literature it has been shown that unless the costs of doing so are too high or too low, consumers or merchants might subscribe to different payment cards (multi-homing) [7]. Furthermore [8,9] show that multi-homing is mostly observed by either consumers or merchants and [10] imply that multi-homing of merchants benefits consumers significantly more than the merchants themselves. [11] points out that the compatibility of payment cards is made less likely in the presence of multi-homing.

A common feature of these models is that they investigate equilibrium outcomes but ignore the dynamic paths towards these equilibria. In addition, the underlying structure of the competition between merchants to attract consumers is in many cases not modeled realistically. To this end we will develop an agent-based model of the payment card market which captures the competition by merchants for consumers as well as the competition between payment cards through decisions for adopting a card by consumers and merchants. Allowing the market to evolve over time we are able to observe the dynamics of our model and determine its properties through computer experiments. By developing a spatial structure in our model we are also able to obtain the origin of multi-homing which cannot be derived in the above models.

The coming section introduces the basic set-up of the model elements and section 3 then describes the interactions between them. Section 4 evaluates the computer experiments of our model and section 5 concludes our findings.

2 Model Elements

Our model consists of three key elements: merchants, consumers and the payment cards. The focus of our investigation will be laid on the interactions between consumers and merchants and the choice which payment cards to hold and accept. We do not consider the interchange fees as a relevant decision variable, but rather treat all fees as exogenously given. A justification for this approach can be deducted from [12] who finds that the degree of competition does not affect the relative fee structure but only the total level of fees, although [13] state that price structures will affect economic outcomes. Thus competition between payment cards will not manifest itself in different interchange fees

between cards and it is therefore reasonable to infer that fixing their level will not affect the outcomes of our paper.

2.1 Merchants

Suppose we have a set of merchants $\mathcal{M}$ with $|\mathcal{M}| = N_{\mathcal{M}}$ who are offering a homogenous good at a fixed common price and face marginal cost lower than this price. Eliminating price competition between merchants allows us to concentrate on the competition between payment cards and how the card choice affects merchants. The merchants are located at random intersections of a $N \times N$-lattice, where $N^2 \gg N_{\mathcal{M}}$. Let the top and bottom edges as well as the right and left edges of this lattice be connected into a torus. Using such a network has the advantage that it approximates more closely than other forms, e.g. random graphs or small-world networks, the geographical location of consumers and merchants.

2.2 Consumers

Consumers occupy all the remaining intersections of the above lattice. The set of consumers is denoted $\mathcal{C}$ with $|\mathcal{C}| = N_{\mathcal{C}}$, where $N_{\mathcal{C}} \gg N_{\mathcal{M}}$ and $N^2 = N_{\mathcal{M}} + N_{\mathcal{C}}$.

Each consumer has a budget constraint that allows him in every time period to buy exactly a single unit of the good offered by the merchants. He will do so only by visiting a single merchant. The utility gained from the consumption of this good exceeds the utility from not buying the good and saving the money for later consumption. Given the common shopping behavior of consumers it would be reasonable to set the length of a time period to be one week. In order to obtain the goods any consumer $c \in \mathcal{C}$ has to travel to a merchant $m \in \mathcal{M}$, where the distance of this travel is measured by the "Manhattan distance" $d_{c,m}$ between the locations on the lattice; the distance between two adjacent intersections is normalized to unity.

This distance imposes travel costs on consumers which reduces the attractiveness of visiting a merchant. As we will show below, consumers prefer card payments over cash payments. Let us for now assume that when deciding which merchant to visit the consumer has not yet decided which of the cards he holds will be used. In this case the more common payment cards the merchant and the consumer have, $\nu_{c,m}$, the more attractive a merchant becomes. From these deliberations we propose to use a preference function for consumer $c \in \mathcal{C}$ to visit merchant $m \in \mathcal{M}_c$, where $\mathcal{M}_c \subset \mathcal{M}$ denotes the set of merchants a consumer considers to go to:

$$\upsilon_{c,m} = \frac{\frac{\nu_{c,m}}{d_{c,m}}}{\sum_{m' \in \mathcal{M}_c} \frac{\nu_{c,m'}}{d_{c,m'}}}. \tag{1}$$

2.3 Payment Cards

There exists a set $\mathcal{P}$ of payment methods with $|\mathcal{P}| = N_{\mathcal{P}} + 1 \ll N_{\mathcal{M}}$. The first payment method is the benchmark and can be interpreted as cash payment while all other payment forms are card payments. In order for a card payment to occur, the consumer as well as the merchant will have to have a subscription to the card in question.

For each unit of the good bought using a payment card, a merchant receives benefits. These benefits may include reduced costs from cash handling. Cash payments do not produce any benefits. Consumers also receive benefits from paying by card, but no benefits from cash payments. The benefits may arise from the delayed payment, insurance cover or cash-back options. For simplicity let us assume that these benefits are identical across cards for all consumers and merchants.

3 Interaction of Elements

The three elements of our model are in constant interaction. The issuers of payment cards are only passive by not making any choices and accepting any subscription request or cancelation, while consumers and merchants make active decisions. The decisions by the merchants are limited to the choice of payment cards they subscribe to. The consumers, on the other hand, have to make decisions affecting the subscription to payment cards, which merchant to choose for their purchase and which payment card to use in a transaction.

3.1 Choice of Merchant

Each consumer $c \in \mathcal{C}$ chooses a merchant $m \in \mathcal{M}_c$ with probability $\upsilon_{c,m}$ as defined in equation (1). The consumers will continuously update their information on the number of common payments, $\nu_{c,m}$ by observing the number of common payments of all merchants they may visit. We restrict consumers to the $N_{\mathcal{M}_c} = |\mathcal{M}_c|$ nearest merchants, i.e. consumers are restricted to visit merchants only in their neighborhood. In the case of several merchants having the same distance from a consumer the merchants are chosen randomly for the length of the experiment.

3.2 Choice of Payment Card

The consumer decides which payment card he wants to use in a transaction with the merchant. Given the benefits associated with the use of the payment cards, he will choose them whenever possible. Let us assume that the consumers select randomly the cards from he holds and the merchants accepts. Only if no such card exists will the transaction be conducted using cash.

3.3 Consumer Subscriptions

Consumers have in every period of time to decide whether to cancel a subscription to a card they hold and whether to subscribe to new cards. Every consumer $c \in \mathcal{C}$ keeps track whether a card he presented has been accepted or rejected. If card $p \in \mathcal{P}$ has been rejected by the merchant, he increases the score $\omega^-_{c,p}$ by one. With ω_c denoting the number of merchants visited, we assume that he cancels his subscription with probability

$$p^-_{c,p} = \frac{\exp\left(\frac{\omega^-_{c,p}}{\omega_c}\right)}{a + \exp\left(\frac{\omega^-_{c,p}}{\omega_c}\right)}, \tag{2}$$

where a accounts for the inertia of consumers to change cards. Similarly on every visit to a merchant he notices that he may accept another card the consumer does not possess, in this case he increases the score of the card he notices, $\omega^+_{c,p}$, by one. The probability of subscribing to this card is then given by

$$p^+_{c,p} = \frac{\exp\left(\frac{\omega^+_{c,p}}{\omega_c}\right)}{a + \exp\left(\frac{\omega^+_{c,p}}{\omega_c}\right)}. \tag{3}$$

It has to be pointed out that $p^-_{c,p}$ denotes the probability of canceling an existing subscription, while $p^+_{c,p}$ is the probability of subscribing to a new card and therefore in general we observe that $p^-_{c,p} + p^+_{c,p} \neq 1$.

We thus observe that widely accepted cards are less likely to be dropped by consumers than less widely accepted cards; similarly will consumers be more likely to subscribe to more widely accepted cards than less widely accepted cards given that they can be used more easily.

This form of decision making by consumers (and likewise by merchants to be presented in the coming section) is not intended to represent a form of learning but merely models the reaction of consumers to their experience of card acceptances (card holdings for merchants) in order to maximize the benefits from their card holdings.

3.4 Merchant Subscriptions

Merchants keep track of all cards presented to them by consumers. Every time a card $p \in \mathcal{P}$ the merchant $m \in \mathcal{M}$ subscribes to is presented he increases the score of $\theta^-_{m,p}$ by one; and if he does not subscribe to the card, the score of $\theta^+_{m,p}$ is increased by one. He decides to subscribe to a new card with probability

$$\pi^+_{m,p} = \frac{\exp\left(\frac{\theta^+_{m,p}}{\theta_m}\right)}{\alpha + \exp\left(\frac{\theta^+_{m,p}}{\theta_m}\right)}, \tag{4}$$

where θ_m denotes the number of cards presented. Similarly he decides to cancel the subscription of a card with probability

$$\pi^-_{m,p} = \frac{\alpha}{\alpha + \exp\left(\frac{\theta^-_{m,p}}{\theta_m}\right)}, \tag{5}$$

where α represents the inertia to changes as before. As with the consumer choice, merchants are more likely to subscribe to cards that are widely held and used than to less widely held cards. Again we note that $\pi^-_{c,p}$ denotes the probability of canceling an existing subscription, while $\pi^+_{c,p}$ is the probability of subscribing to a new card and therefore in general we observe that $\pi^-_{c,p} + \pi^+_{c,p} \neq 1$.

The ideal situation for consumers and merchants would be if all agents were able to coordinate their subscription decisions and agree on a single payment card. We do not provide a formal mechanism for such a coordination of choices but the way card subscriptions are chosen decentrally will nevertheless result in a strong coordination

as we will show below. It are the benefits from coordination rather than a desire to coordinate that drive the dynamics through a constant search process.

4 Computer Experiments

Using the above model we conducted computer experiments using the following fixed parameters settings: $N_{\mathcal{C}} = 1100$, $N_{\mathcal{M}} = 125$, $N_{\mathcal{P}} = 9$, $N_{\mathcal{M}_c} = 5$, $a = 8$, and $\alpha = 15$. We investigated the model over 1,000 time periods, i.e. the equivalent of approximately 20 years, and found that other parameter constellations yield similar results.

Initially each consumer and merchant is allocated a random number of cards between zero and nine with equal probability and then obtains each card with equal probability. We also set $\theta^{-}_{m,p} = \theta^{+}_{m,p} = \omega^{-}_{m,p} = \omega^{+}_{m,p} = 0$. Furthermore, throughout the experiment consumers and merchant have the ability to make and accept cash payments if no card payment is possible.

4.1 Market Shares

Our experiments using the above model reveal that the market quickly becomes dominated by a small number of cards as the top left panel in figure 1 shows. The dominance can be measured in terms of the fraction of transactions conducted using a specific card, the fraction of consumers holding the cards as well as the fraction of merchants accepting it. In nearly all cases we observe that card payments drive out cash transactions nearly completely and only two cards survive in the long run.

What we also observe from our experiments is the importance of the initial market share. We find that those cards with the largest market share in the initial random distribution of cards are those dominating the market over time, with the order maintained. We see furthermore that two or more rarely three cards coexist for a long period of time, where the relative market shares are not changing significantly over time. Three or more cards only survive for a considerable time in cases where the initial market shares are sufficiently identical and large.

We find evidence that very slowly over time only the largest card will survive and thus obtain a monopoly. This process however is very slow and takes considerable time, well beyond the 1000 time periods investigated here. The reason for this very slow development can be found in the multi-homing we consider next.

4.2 Multi-homing

The second important result we obtain from our experiments is that consumers usually hold a single card; only a small fraction of consumers actually hold two cards and very rarely three cards after the market has settled. In contrast to this, a sizable number of merchants accept two or even three cards, even after many time periods have elapsed, see top right and bottom left panel in figure 1. The reason for this observation is that merchants serve a large customer base and thus more easily come into contact with consumers holding different cards, providing incentives for them to accept more than one card to increase their benefits from accepting card payments. This result is very

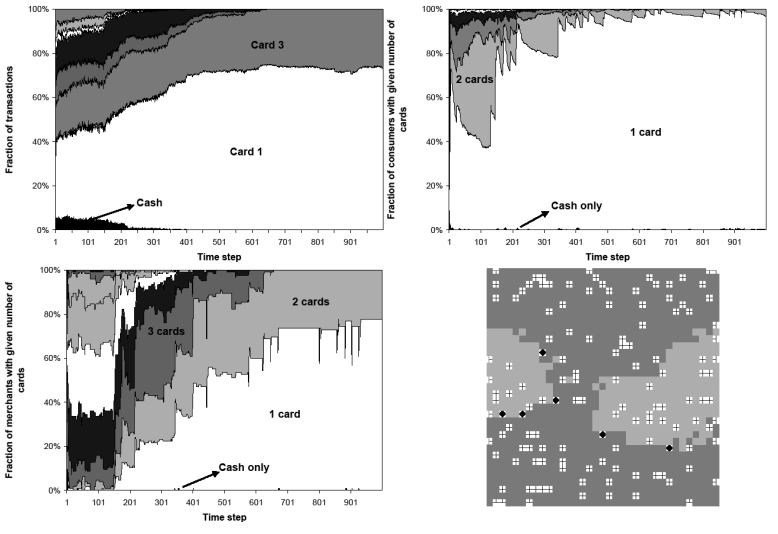


Fig. 1. Market share of individual payment cards (top left), multi-homing of consumers (top right) and merchants (bottom left) and regional use of payment cards by consumers after 1000 time steps (bottom right); different shades denote different cards, empty cells the location of merchants and a rhombus the use of cash

much in agreement with the outcomes of the models in [8,9] who in a very different setting also suggest that multi-homing will mostly be observed by either consumers or merchants. We find here with the above reasoning that merchants engage much more in multi-homing than consumers.

As we see from figure 1, card usage of any surviving card is concentrated in distinct geographical zones which are generally not overlapping. Those merchants located nearest to the edges of these zones tend to accept multiple cards in order to obtain benefits from all consumers wishing to use a card for payment. A small number of cash transactions occur in those cases where the merchant did not accept the card of the consumer. The existence of these geographical boundaries is responsible for the emergence of multi-homing, an effect the equilibrium models in the literature have thus far not been able to establish.

The willingness of merchants to subscribe to multiple cards is responsible for the fact that once the number of cards in the market is reduced to two or three they tend to co-exist for a considerable period of time. It is also not surprising that merchants are subscribing to multiple cards as the large number of transactions they conduct in every time period makes benefits of multi-homing much more important than for consumers. Given the multi-homing of merchants, consumers no longer have an incentive to hold multiple cards.

This result is not in contrast to [10] who find that most of the benefits of multi-homing by merchants goes to consumers. Although consumers obtain the majority of benefits from the multi-homing of merchants by not having to subscribe to multiple cards or relying on non-beneficial cash transactions, the large number of consumers

each merchant interacts with, provides him with sizeable benefits of multi-homing. These larger benefits to each individual merchant induces him to subscribe to multiple cards, a situation similar to the production of public goods.

Although the network externalities reduce the number of cards in the market quickly, it is the multi-homing of merchants that prevents a single card from obtaining a monopoly quickly. Thus competition between cards is sustained and the subsequent dominance of a single card is limited due to multi-homing.

4.3 Discussion

On first sight the above results on multi-homing seem not to be realistic as undoubtedly a large number of consumers have more than two credit cards in their possession. It has, however, to be noted that this in part refers to cards issued by different financial institutions rather than different card organizations, such as Visa or Mastercard. As any of these cards are accepted equally, we treat them here as a single card in our model. This aggregation of cards in the real world would reduce the number of cards held significantly and make our result much more realistic. Furthermore, [14] finds empirically that although consumers may possess several different cards, they in many cases only use one of these cards regularly, which would be very much in line with the results from our model.

Another source of payment cards are store cards which are only accepted by the issuing store. We excluded such cards from our model as they cannot be used with different merchants and thus do not have the same network effects. They could nevertheless present competition to payment cards.

In light of the above comments we can observe that the results we obtained are at least approximately realistic. Despite a competitive market with cards not differing from each other we observe the coexistence of two or three credit cards dominating the market very quickly, while at least for some time at the beginning of the experiments a number of less important credit cards survives. As this reflects quite well the current market situation noted in the introduction, our model suggest that we should expect even more consolidation of the market in the future.

The parallel existence of two or more cards is equivalent to the equilibrium [15] obtain in their model when the costs of substitution between networks, i.e. changing subscriptions, is low relative to the network externality. Given the multi-homing of merchants the costs for consumers of switching cards is negligible, fulfilling the conditions for this equilibrium. Given the geographical zones it would be on the other hand very costly for merchants to give up either multi-homing close to the boundaries or switch cards within the zones.

The main deviation from reality is the emergence of very distinct geographical zones of card usage which is clearly not replicated in reality. Although differences exist in the market shares of payment cards across countries, they are significantly less pronounced than in our model. [14] finds a correlation between the cards held by consumers and acceptance by merchants, leading to a certain degree of regional concentration of card usage through a local interaction loop, but the regional differences are by no means as distinct as we find in our model.

However, we did only allow consumers to visit the five nearest merchants and ignored any consumers traveling longer distances, e.g. for consumers being on holidays or business journeys. Including such long-distance travel into our model, such as the inclusion of small-world effects in the network structure, might actually change the results of our model and need to be explored in more detail. It is a widely accepted fact that the network topology can have significant impact on the outcomes of any dynamics using its structure. [16] provides an example for the different outcomes of a repeated prisoner's dilemma played on various types of networks and it is reasonable to expect similar differences in our model.

5 Conclusions

We provided an agent-based model of the market for payment cards where the interaction between merchants and consumers drove the decision to subscribe to a payment card and subsequently the market shares. We found that the market quickly becomes dominated by a small number of payment cards operating in distinct geographical markets and multi-homing of merchants emerged.

This novel approach to model the payment card market yielded additional insights into the dynamics of the market which conventional models with their emphasis on equilibrium outcomes cannot provide. It became clear that the multi-homing of merchants enables the survival of a small number of payment cards, despite the presence of network externalities. The emergence of multi-homing prevents the quick appearance of a monopolistic market with only a single card surviving, thus preserving competition between cards.

A large number of extensions can be thought of to improve the model further and make it more realistic. The importance of the information set on which consumers and merchants base their decisions has already been shown in [17]. As was already mentioned above it would further be of interest to evaluate the importance of the network topology for the results in our model. Enabling payment cards to react to changes in their market shares through giving different net benefits to consumers and merchants as an incentive to subscribe to the card and actually use it, might provide further insights into the dynamics of the payment card market. The addition of fixed fees for card subscriptions and finally an interchange fee would further complete the model.

References

1. Schmalensee, R.: Payment systems and interchange fees. Journal of Industrial Economics 50, 103–122 (2003)
2. Rochet, J.-C., Tirole, J.: Cooperation among competitors: Some economics of payment card associations. RAND Journal of Economics 33(4), 549–570 (2002)
3. Wright, J.: Pricing in debit and credit card schemes. Economics Letters 80, 305–309 (2003)
4. Markose, S., Loke, Y.J.: Network effects on cash-card substitution in transactions and low interest rate regimes. Economic Journal 113(487), 412–456 (2003)
5. Rochet, J-C., Tirole, J.: Platform competition in two-sided markets. Journal of the European Economic Association 1(4), 990–1029 (2003)

6. Guthrie, G., Wright, J.: Competing payment schemes. National University of Singapore Department of Economics Working Paper No. 0311 (2003)
7. Goyal, S., Janssen, M.C.W.: Non-exclusive conventions and social coordination. Journal of Economic Theory 77, 34–57 (1997)
8. Gabszewicz, J.J., Wauthy, X.Y.: Two-sided markets and price competition with multihoming. Mimeo, CORE (2004)
9. Roson, R.: Platform competition with endogenous multihming. Fondazione Eni Enrico Mattei Note di Lavoro 20.2005 (2005)
10. Chakravorti, S., Roson, R.: Platform competition in two-sided markets: The case of payment networks. Federal Reserve Bank of Chicago Working Paper (September 2004)
11. Doganoglu, T., Wright, J.: Multi-homing and compatibility. National University of Singapore Department of Economics Working Paper No. 0314 (2003)
12. Gonçalves, R.: Policy challenges in two-sided network industries. Europe Economic Working Paper (2003)
13. Rochet, J.-C., Tirole, J.: Two-sided markets: An overview. Mimeo, GREMAQ, Toulouse (2004)
14. Rysman, M.: An empirical analysis of payment card usage. Mimeo, Boston University (2004)
15. Mitchell, M.F., Skrzypacz, A.: Network externalities and long-run market shares. Stanford Graduate School of Business Research Paper No. 1879 (2005)
16. Wilhite, A.: Economic activity on fixed networks. In: Tesfatsion, L., Judd, K. (eds.) Handbook of Computational Economics, vol. II, North-Holland, Amsterdam (2006)
17. Kabadjova, B.A., Tsang, E., Krause, A.: Competition among payment cards: An agent-based approach. Mimeo, CCFEA Working Paper, University of Essex, Great Britain (2005)

The Possibility of an Epidemic Meme Analogy for Web Community Population Analysis

Masao Kubo[1], Keitaro Naruse[2], Hiroshi Sato[1], and Takashi Matubara[1]

[1] National Defense Academy of Japan, Dep. of Computer Science, Hashirimizu 1, Yokosuka, Kanagawa,239-8686, Japan
{masaok,hsato,matubara}@nda.ac.jp
[2] Univ. of Aizu, Dep. of Computer Software, Aizu-Wakamatsu, Fukushima-ken, 965-8580, Japan
naruse@u-aizu.ac.jp

Abstract. The aim of this paper is to discuss the possibility of understanding human social interaction in web communities by analogy with a disease propagation model from epidemiology. When an article is submitted by an individual to a social web service, it is potentially influenced by other participants. The submission sometimes starts a long and argumentative chain of articles, but often does not. This complex behavior makes management of server resources difficult and a more theoretical methodology is required. This paper tries to express these complex human dynamics by analogy with infection by a virus. In this first report, by fitting an epidemiological model to Bulletin Board System (BBS) logs in terms of a numerical triple, we show that the analogy is reasonable and beneficial because the analogy can estimate the community size despite the submitter's information alone being observable.

Keywords: Kermack–McKendrick models, SIR, BBS, SNS, Web Mining.

1 Introduction

Service industry is rapidly growing. Abe of Fujitsu [1] says, ''As a result, various services that were previously processed, handled, and separated as merely mass services have now become possible to provide as individually targeted and personalized services''. The shift of this quality of service requires more scientific approach to service system. For example, in these service systems, it is difficult to manage the system resource based on the intuition and experience because a huge number of different services have to be involved.

We believe that resource allocation for Social Networking Service (SNS), public Bulletin Board Systems (BBSs), and other social applications is a service that should be researched, as stated above. For a server manager who must allocate resources adequately to provide comfortable services, estimation of the number of accesses in the near future is an important problem. In this paper, we are interested in estimation of size of access in near future when a time series data of their access log is observable because we think it makes its job easier.

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1073–1080, 2007.

This is currently performed as a regression analysis of time series data for the number of accesses. However, this method is inadequate when a web service offered belongs to Web 2.0 because customers interact with each other. For example, even a small community produces many accesses, so that the interaction among members is complex. Therefore, the population size of the community and the characteristics of its members are also important in making precise predictions.

This observation suggests using the model for an epidemic. An appropriate epidemiological model describes the dynamics in terms of three groups: Susceptible, Infected, and Recovered (SIR). The numbers in the SIR groups are described by differential equations having two constants, β and γ, which represent infection and recovery speed, respectively (details are described in Section 3). Because there are no actual contagious diseases when connected to the Internet, this approach may seem unnatural. However, consider supposing that a *meme* spreads among people who meet via web services? This is a popular idea, as we describe in the next section, but as far as we know, there is little research that uses actual logged access data. If the behavior of web service communities is described by the epidemiological model, we think this can be a powerful tool for server management, which usually depends on experience, because the population of the community is $(S + I + R)$ and the population of the three subgroups can be estimated precisely.

This paper reports on some of the initial work. First, numerical verification must be performed. It will be a weak argument if there are big gaps between the time series of the population of the epidemiological model and real server access data, even if the epidemiological model explains human behavior around web services well from a semantic viewpoint. Therefore, in this paper, by showing some curve fitting results for the actual log data of a TV-related BBS, we conclude that this approach is plausible.

This paper is organized as follows. In the next section, related work about population predictions for web services is summarized. The epidemiological model used here is explained in Section 3. In Section 4, an analogy between human behavior in a web community and the epidemiological model is proposed. Then, the parameters of the model are estimated by curve fitting in Matlab. At the end of Section 4, we discuss our next step: agent-based simulation.

2 Background

If you are managing a web site, you may want to know the number of accesses to your site in the future. Initially, the number of accesses per unit time is calculated by the methods in [2] and [6] for example, because real log data is noisy. Next, by using this smoothed time series data, the number of future server accesses is estimated.

The simplest estimation method is regression analysis. A type of distribution, such as an exponential distribution and Gumbel distribution, is assumed a priori. This is simple, but it does not consider why people are accessing the service. For further analysis, a model is required that can explain why and how people are accessing.

Gruhl et al. [4] listed two candidate models for such human behavior, innovation propagation dynamics and disease propagation. They adopted the innovation

propagation model because of their objective of time-order reconstruction of SNS sites. Therefore, the latter model was not examined.

We also think the latter disease propagation model from epidemiology is as reasonable a candidate as the dynamics of knowledge propagation. If people are interested in an event, such as the press release of a new book, they will continue to search and make notes in their web pages, SNS, BBS, etc. Therefore, this will increase the chance that someone will notice the event. This seems similar to infection by a disease. Then, with the proliferation of such pages and other descriptive material, effective pages that offer infected people convincing arguments will emerge. People who meet such definitive pages will stop their search and discussion of the event. We think that the above process mimics the infection/recovery dynamics of disease propagation models. Therefore it is reasonable to ask if this propagation model is appropriate for explaining human behavior in the web community.

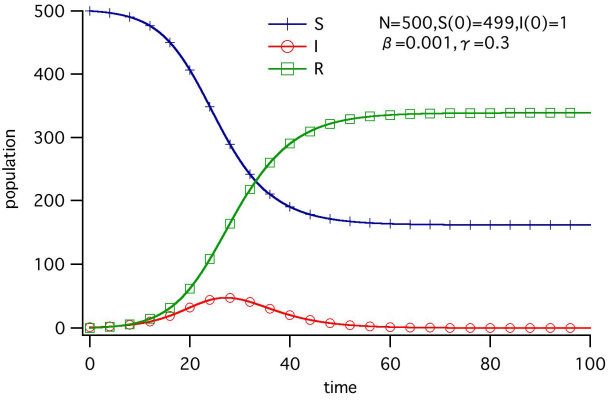


Fig. 1. An example of an SIR model. N = 500, β = 0.001, γ = 0.3, γ/β = 300.

As we noted in Section 1, this idea is not new. In particular, as in the survey in [4], there is much research from the network analysis point of view. For example, the network of migration is important for the accurate prediction of an outbreak of disease when people go to a city centre to work and then go back home. In the case of sexually transmitted diseases, the network of sexual activity is very important for suppression of the outbreak, because the number of sexual activities of a person follows a power law. However, as far as we know, there is no work that applies an epidemiological model based on network theory to actual communication data in the Internet community, except for computer virus analysis.

3 The Disease Propagation Model in Epidemiology

3.1 The Kermack–McKendrick Model

There are many disease propagation models in epidemiology because of the different propagation conditions for diseases. However, the Kermack–McKendrick model

(1927) and the Reed–Frost model (1928) [5] are much simpler and more general than others.

The Kermack–McKendrick SIR model gives the differential equations for a deterministic general epidemic [7]. Let

$$S + I + R = N, \tag{1}$$

where S, I, and R are the number of Susceptible, Infected, and Recovered people, and N is constant. S(t), I(t), and R(t) are represented as follows:

$$dS/dt = -\beta S \cdot I, \tag{2}$$

$$dI/dt = \beta S \cdot I - \gamma I, \tag{3}$$

$$dR/dt = \gamma I, \tag{4}$$

where β is the infection probability and γ is the recovery probability. Clearly, there is no direct transition from S to R. From the epidemiology point of view, β is the number of people who are infected by a patient and it is necessary for $1/\gamma$ unit time on average until a disease is cured [10]. Usually, in the case of person-to-person disease infection, these parameters are estimated statistically. When β and γ are available, the important information is given as follows. All the members of S are not always infected (see Fig. 1). The condition that an epidemic ends is given by $dI/dt = 0$. The solution is

$$I = 0 \text{ or } S = \gamma/\beta. \tag{5}$$

In addition, as $R(0) = 0$, the number of people who were not contagious, $S(\infty)$, is satisfied as follows:

$$S(\infty) = S(0)\exp(-(N - S(\infty))/(\gamma/\beta)). \tag{6}$$

3.2 The Network and Epidemic Model

Research using epidemiological models based on networks usually assumes a degree distribution. In our case, the propagation path for each member of the community is usually unknown and different. Therefore, for simplicity, we assume a fully mixed model [9].

4 The Analogy and Experiments

4.1 Data

Compared with 10 years ago, it is now easy to collect data on the behavior of human groups because of the development of web-crawling agent technology and social networking services. There are various communities with different cultures and subjects of interest. We think that it is important to choose the largest communities possible, to maximize the generality of this discussion.

Therefore, we looked to BBSs (aka Internet forums) such as Slashdot and Google Groups, for data to analyze. As is widely known, BBSs on the Internet are social network services that offer the chance of communication and discussion only. When an individual submits an article to a BBS, it is influenced by other users of the BBS.

We chose the biggest Japanese open anonymous BBS, "2 channel" (http://www.2ch.net/). This BBS includes more than 600 categories, and each category contains from 100 to 500 threads. It processes more than 100 million page views per day. Anyone, without special privileges, can access the same data that we acquired, and this site is frequently used by other web researchers in Japan. We would expect that this is the most widely accessed BBS under present conditions.

For time series analysis using the SIR model, the start time is important. However, this BBS is available 365 days per year, and anyone can post an opinion at any time. In fact, it is difficult to specify when a discussion actually starts. Gruhl [3][4] identified two behavior types for SNSs by their cause: *spike* or *chat*. Spike refers to bursty behavior by events outside the community. By contrast, chat is a burst based on a conversation within the community. For our purposes, we would like to pick a spike at the time when a known event happens. Therefore, we adopted a TV program and its related BBS where participants talk to each other even outside the broadcast time. However, we know beforehand when the maximum external stimulus will happen, from newspaper information. In addition, we can suppose that there will be no spike following the broadcast.

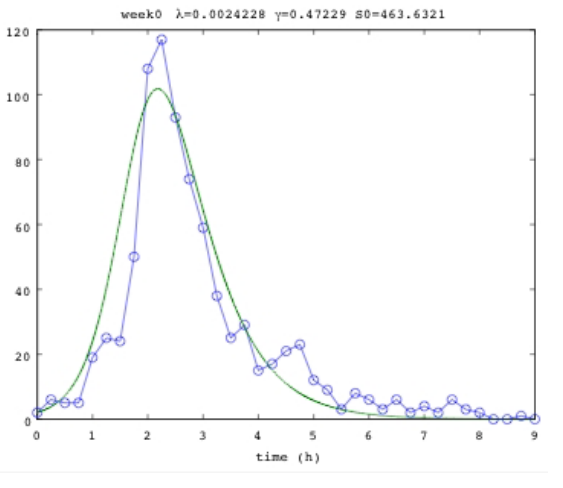


Fig. 2. Posting data for the BBS (Jan. 10 to Jan. 11, 2007) and the fitting result for the SIR model

4.2 The Proposed Interpretation of BBS Data by the SIR Model

Here, we assume that "Susceptible" means a person who is interested in the TV program. A person who has so strong an opinion as to post to the BBS corresponds to "Infected". A "Recovered" person leaves the BBS, being no longer interested in the

topic. Therefore, we aim to minimize the RMS difference between the log data of the number of posters and the "Infected" group, as calculated using (3).

4.3 Fitting and Results

We counted the number of posters every 15 minutes from 9 pm, Jan. 10, 2007 to 6 am, Jan. 11, 2007. It was 10 pm when the TV program broadcast started, and it finished at 11 pm. This TV program was so famous and general as to be watched by over 18% of households in Japan. The zigzag line of Fig. 2 indicates the logged data. The x-axis indicates the time sequence and each tick is an hour. In this figure, the TV program starts at x = 1 and ends at x = 2. Note that there is a big burst with a small drop around x = 1.5. We think this is reasonable because posters will also want to watch the TV program!

The smoother line of Fig. 2 represents the fitting result of the SIR model using Matlab. It seems that this fits well. The resulting estimate for the triple (S(0), β, γ) is (463.6321, 0.0024228, 0.47229).

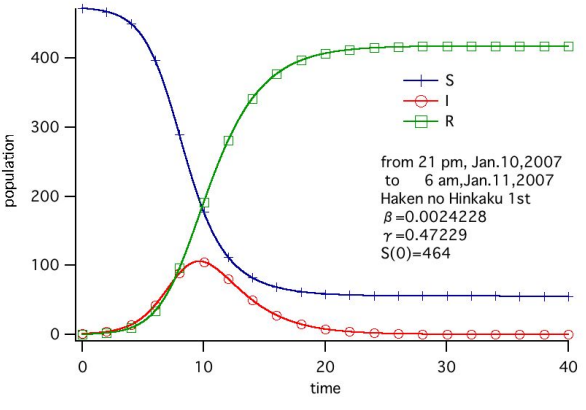


Fig. 3. The resulting behavior of SIR (Jan. 10 to Jan. 11, 2007)

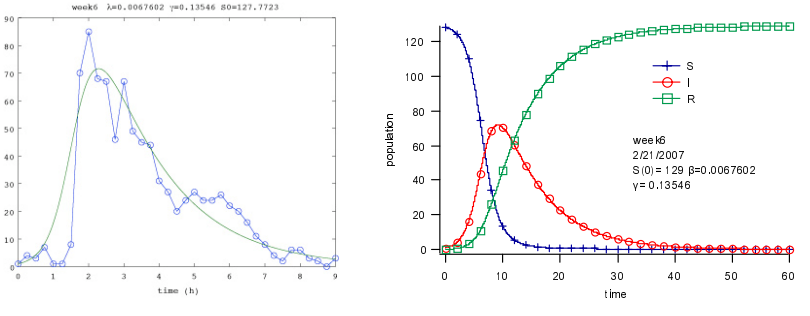


Fig. 4. The fitting results for another broadcast (Feb. 21 and Feb. 22, 2007)

Fig. 3 shows the progress of S, I, and R. The I in Fig. 3 is the same as the solid line of Fig. 2. If our assumption of Section 4.2 is valid, this offers the following insights about the community: (1) about 464 persons came to this BBS, (2) about 400 people left, and (3) 60 people still enjoy the community. This third point suggests that the broadcast and the surrounding discussion in this BBS enlarged its community by about 60 people.

Fig. 4 shows the results for another week (from 9 pm, Feb. 21, 2007). This curve also fits well to the week's data. The estimated γ is smaller than that for the earlier week, with posts continuing longer than for the Jan. 10 to 11 period.

4.4 Discussion

In this paper, we validated the disease propagation model as a model for explaining human behavior in a community. By using curve-fitting techniques, we report how reasonable this idea is.

We adopted the Kermack–McKendrick version of the disease propagation model and applied it to the posting data of a BBS. The characterizing triple $(S(0), \beta, \gamma)$ for this model was estimated by RMS minimization and hill climbing in Matlab. As shown in Figs. 2, 3, and 4, the SIR model fitted well and gave new and insightful information.

This approach has the following attractive aspects. Firstly, this model can estimate the total community size, namely $(S + I + R)$, which regression analysis via a statistical distribution cannot achieve. Secondly, it is easy to understand the behavior intuitively, with the propagating speed being β, and the durability of conversation seeming to be γ. We hope it will ease the management of server resources, with (5) and (6) describing the population dynamics of each community.

Note that, in Section 4.2, we proposed understanding the logged data of BBSs in terms of an epidemiological propagation analogy. That is, we assumed that people join the BBS so as to post. In actual BBS communities, there are many "lurkers", and [8] estimates the total community size including lurkers. However, we think the analogy remains reasonable because the driving force of a BBS is undoubtedly the group of people who want to post.

The unique point of this analogy is that it deals with migration among communities. S and R represent the population outside the observing BBS community. We think these estimates justify evaluating the nonlinear differential equations because this information is expensive even if you can obtain access to it.

The proof that the assumptions above are correct is very important and an urgent task for us. We think that Agent-Based Simulation (ABS) is a powerful tool for SSME [11]. As is well known, one definition of "Engineering" is that it is a methodology for obtaining desirable results. However, in a service industry, it is sometimes difficult to test a new method. In such cases, the agent simulation approach is one that is both practicable and meaningful. It is possible for this bottom-up simulation methodology to use a set of programs that behave like participants using the service. As tastes vary, we could use a variety of agents, with carefully chosen parameters. In our case, verification of posting article behavior of clients is required because this component is out of focus of any disease propagation model.

5 Conclusion

In this paper, we have proposed a new approach to understanding the behavior of the Internet community by analogy with a disease propagation model from epidemiology. The SIR of the Kermack–McKendrick model was applied to data comprising the number of posts per 15 minutes to a BBS. The characterizing (S(0), β, γ) of this model was estimated by RMS minimization and hill climbing in Matlab. This new interpretation fits well, and we can say that the analogy is a promising approach that gives new and insightful information, namely:

(1) A response from a big event for community is represented by infection speed, recovery speed, and initial community size.
(2) This framework offers information about community migration. In particular, the total community size is highly valuable information for server managers of web service sites, who have to allocate resources.

References

1. Abe, T.: What is Service Science Dec-05, FRI Research Report, No.246 (2005)
2. Fujiki, T., Nanno, T., Suzuki, Y., Okumura, M.: Identification of Bursts in a Document Stream. In: First International Workshop on Knowledge Discovery in Data Streams (in conjunction with ECML/PKDD 2004) (2004)
3. Gruhl, D., Guha, R., Kumar, R., Novak, J., Tomkins, A.: The Predictive Power of Online Chatter. In: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 78–87. ACM Press, New York (2005)
4. Gruhl, D., Guha, R., Liben-Nowell, D., Tomkins, A.: Information Diffusion through Blogspace. In: Proceedings of the 13th International Conference on the World Wide Web, pp. 491–501. ACM Press, New York (2004)
5. Jacquez, J.A.: A Note on Chain-Binomial Models of Epidemic Spread: What is Wrong with the Reed–Frost Formulation? Mathematical Biosciences 87(1), 73–82 (1987)
6. Kleinberg, J.: Bursty and Hierarchical Structure in Streams. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2002)
7. Krebs, C.J.: Ecology. Benjamin-Cummings Publishing Company (2002)
8. Naruse, K., Kubo, M.: Lognormal Distribution of BBS Articles and its Social and Generative Mechanism. Web Intelligence, 103–112 (2006)
9. Newman, M.E.: Spread of Epidemic Disease on Networks. Physical Review E 66, 016128 (2002)
10. Nishiura, H., Inaba, H.: Prediction of Infectious Disease Outbreaks with Particular Emphasis on the Statistical Issues Using a Transmission Model. In: Proceedings of the Institute of Statistical Mathematics, vol. 54(2), pp. 461–480 (2006) (in Japanese)
11. Rahmandad, H., Sterman, J.: Heterogeneity and Network Structure in the Dynamics of Contagion: Comparing Agent-Based and Differential Equation Models, MIT Sloan School of Management, Cambridge MA 02142 (2004)

The Econometric Analysis of Agent-Based Models in Finance: An Application

Youwei Li[1], Bas Donkers[2], and Bertrand Melenberg[3]

[1] Queen's University Management School, Queen's University of Belfast, BT7 1NN, Belfast, United Kingdom
y.li@qub.ac.uk
[2] Department of Business Economics, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, The Netherlands
donkers@few.eur.nl
[3] Department of Econometrics and O.R., Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands
b.melenberg@uvt.nl

Abstract. This paper illustrates how to compare different agent-based models and how to compare an agent-based model with real data. As examples we investigate ARFIMA models, the probability density function, and the spectral density function. We illustrate the methodology in an analysis of the agent-based model developed by Levy, Levy, Solomon (2000), and confront it with the S&P 500 for a comparison with real life data.

1 Introduction

Many of the classical models in finance are based upon the assumptions of investor homogeneity and expected utility theory, with as prominent example the Capital Asset Pricing Model (CAPM), see, for example, Cochrane (2001). Although recent research reveals that the assumptions underlying the CAPM are hard to maintain (see Barberis *et al.* (2001) for a recent discussion), they are still used because of their analytical tractability.

The developments of computational power provide the opportunity to relax these assumptions through the use of simulation techniques. In particular, agent-based models of financial markets have been introduced. These models investigate the interaction of heterogeneous agents, having, for example, different attitudes towards risk or different expectations about future asset returns, and have been studied intensively, see, for example, He and Li (2007a, 2007b), Hommes (2006), LeBaron (2006), Levy *et al.* (2000), Lux and Marchesi (1999), and references therein. So far, this research has mainly focused on investigating whether a single or some 'representative' simulation runs of an agent-based model shares some important characteristics found in actual financial markets, the so-called stylized facts, such as excess volatility, volatility clustering, or fat tails; see Hommes (2006), for instance. These works provide various ways to explain the stylized facts, but typically focus on different sets of stylized facts. In

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1081–1091, 2007.

addition, they match these stylized facts to different degrees. To investigate the relative merits of the various approaches, systematic procedures to investigate the differences between two agent-based models or to judge whether an agent-based model is realistic are needed. Such methods have been developed in Li *et al.* (2006a, 2006b).

The aim of this paper is to illustrative how to use econometric techniques to thoroughly analyze the agent-based models. Although preferably one would like to be able to estimate agent-based models, this is often infeasible. The most important reason is the complexity of the agent-based models, which makes verification of (econometric) identification rather difficult, and thus proving consistency of estimation troublesome. Moreover, in case estimation is possible, the likely heavily nonlinear relationship between observables and unknown parameters to be estimated might seriously complicate estimation. So, only few attempts have been made to estimate agent-based models, see Alfarano *et al.* (2005) and Boswijk *et al.* (2007) for examples.

Therefore, in this paper, we constrain our study to the comparison of different agent-based models and on how to compare an agent-based model with actual data. We investigate the comparison of the characteristics that are linked directly to some well-known stylized facts: the ARFIMA parameters, the probability density function, and the spectral density function. Other features of the return process have been investigated in He and Li (2007a, 2007b).

A special feature of agent-based models is that its outcome series can be observed along two dimensions, namely, we can observe the outcome series for each run of the simulation, and we can run the agent-based model independently many times. We shall exploit this possibility heavily in this paper when applying econometric techniques, particularly, when comparing actual data with outcomes of agent-based models.

Throughout this study, we will illustrate the methodology using an early example of the agent-based models: the model developed by Levy *et al.* (2000) (LLS model from now on). The LLS model applies a standard economic utility maximization approach for the derivation of market activities of individual investors which is augmented by stochastic factors being borrowed from physics. The LLS model is successful in generating very spectacular crashes and upheavals of market prices. Furthermore, varying combinations of time horizons among traders lead to diverse and sometimes surprising results for the time development of the share of wealth owned by different groups of investors. However, a systematic study of the model, such as sensitivity analysis, and a comparison of the model with actual data has never been performed.

The next section briefly introduces the LLS model and its original parameter settings. We then provide a sensitivity analysis and a comparison of two different agent-based models. This is followed by a comparison of the LLS model to real life data, represented by the Standard & Poor 500 index (hereafter S&P 500). We end with some concluding remarks.

2 The Levy-Levy-Solomon Model

In the LLS model, Levy *et al.* (2000), there are two assets: a stock and a bond. The bond is assumed to be a risk free asset, while the stock is a risky asset. The bond is exogenous with infinite supply, so the investors can buy from it as much as they wish at a given rate of return, r. The stock is in bounded supply. There are N outstanding shares of the stock. The return on the stock is composed of capital gains and dividend payments. The dividend per share at time t, D_t, is a random variable that follows a multiplicative random walk: $D_t = D_{t-1}(1+z)$, where z is distributed uniformly in the range $[z_1, z_2]$. The overall rate of gross return on the stock in period t, R_t, is then given by $R_t = (P_t + D_t)/P_{t-1}$ where P_t is the stock price at time t.

The investors are expected utility maximizers, characterized by the utility index $U(W) = W^{1-\alpha}/(1-\alpha)$, which reflects their personal preference. The investors are divided into two groups, the first group will be referred to as the rational informed investors (RII), and the second group will be referred to as the efficient market believers (EMB).

The RII investors—At time t the RII investors believe that the convergence of the price to the fundamental value will occur in period $t+1$. Furthermore, RII investors estimate the next period fundamental value of stock price P^f_{t+1} by $P^f_{t+1} = E_{t+1}[D_{t+2}]/(k-g)$ according to Gordon's dividend stream model. Here, k is the discount factor, and g is the expected growth rate of the dividend, i.e., $g = E(z)$, which is known to the investors. Using $E_{t+1}[D_{t+2}] = D_{t+1}(1+g)$ and $D_{t+1} = D_t(1+z)$, RII investors thus believe that $P_{t+1} = P^f_{t+1}$ is given by

$$P^f_{t+1} = D_t(1+z)(1+g)/(k-g).$$

Investing a proportion x of wealth in the stock at time t, the expected utility becomes $E\{U(W_{t+1})\} = E\{U(W_t[(1-x)r + xR_{t+1}])\}$, with W_t the wealth at time t, and where the expectation is over the rate of return $R_{t+1} = (P_{t+1} + D_{t+1})/P_t$, with $P_{t+1} = P^f_{t+1}$. A solution for this optimization problem can be found by solving the first order conditions.

The EMB investor—EMB investor i has only a limited memory, and uses the most recent m^i returns on the stock to estimate the ex ante distribution. At time t, each of these past returns on the stock R_j, $j = t, t-1, ..., t-m^i+1$ is given an equal probability $1/m^i$ to reoccur in the next period $(t+1)$. Therefore, the expected utility of EMB investor i is given by

$$E\{U(W_{t+1})\} = \frac{1}{m^i}\sum_{j=1}^{m^i}\frac{1}{1-\alpha}[W_t[(1-x)r + xR_{t-j}]]^{1-\alpha}.$$

Maximization of this expected utility yields the optimal proportion of wealth, x^{*i}, that will be invested in the stock by EMB-investor i. To allow for noise around the optimal portfolio choice, LLS assume $x^i = x^{*i} + \varepsilon^i$ where ε^i is a random variable drawn from a normal distribution with mean zero and standard deviation σ. For simplicity, noise is only added to the portfolio share of stocks for the EMB investors.

Given the stock demand of the RII- and EMB-investors, together with the total supply of N fixed shares, the (temporary) Walrasian equilibrium stock price at time t, P_t, can be determined. This price leads to updated expectations and a new equilibrium arises in the next period, and so on.

The parameters and initial conditions chosen by Levy *et al.* (2000) are as follows: $W_0 = \$1000$, $P_0 = \$20.94$, $D_0 = \$0.5$, $r = 1\%$, $k = 4\%$, $z_1 = -7\%$, $z_2 = 10\%$, $\sigma = 0.2$, $\alpha = 1.5$. Given quarterly trading period, there are 1,000 investors, with 96% RII investors and 4% EMB investors with memory span 5 and 15. This defines the benchmark economy.

3 Sensitivity Analysis

In this section we investigate the sensitivity of the benchmark LLS model to changes in initial conditions and parameter values. We illustrate this by estimating the ARFIMA$(0, d, 0)$-process.

We illustrate the sensitivity analysis in terms of the initial price. The subsequent sensitivity analysis in terms of other initial conditions is performed in a similar way. We first simulate the benchmark model. Next, we keep all the conditions and parameters the same, except for the initial price. Two additional simulations are performed, one with an initial price $P_0 = 26$ that is higher than the benchmark price, and an other one with a lower initial price, namely $P_0 = 16$. Then we look at the impact of these different initial prices on the ARFIMA parameters, and see whether they significantly deviate from those of the benchmark model. Next, we do the same exercises in terms of the initial dividend (with $D_0 \in \{0.4, 0.6\}$), the risk aversion parameter (with $\alpha \in \{1.45, 1.55\}$), the maximal one period dividend decreases (with $z_1 \in \{-0.08, -0.06\}$), and the initial wealth. In the latter case we consider two variations: In the first case (unif.), the initial wealth is uniformly distributed over [500, 1500], while in the second case (50%) half of the investors have an initial wealth of 500 and the other half have of 1500. In this study, for each set of parameter, we ran 5,000 independent simulations over 1,000 time periods, and for each run of the model we use the last 152 observations to wash away the initial noise effects, and to match the sample size of the S&P 500 that we use later on.

Table 1 summarizes the maximum likelihood estimates of the ARFIMA model for the benchmark LLS model, by presenting the average results over the simulations, as well as the numbers of significant parameter estimations. We find on average a negative value of d, which is significant in only 7.6% out of the 5,000 simulations. So, based on a single simulation, we would accept most of the times the hypothesis $d = 0$ of no fractional integration. However, combining the 5,000 simulations, we find a t-value of $t = -23.80$, so that we clearly have to reject the hypothesis $d = 0$! As could be expected, the use of more simulations can thus substantially increase the power of the tests that are performed.

For different initial conditions and parameter settings, we report the estimates of the ARFIMA $(0, d, 0)$ model in Table 2. In Table 3 we report the t-test for the difference of estimated d between the benchmark model and the models with

Table 1. Estimates of ARFIMA$(0, d, 0)$ for the benchmark economy

	Coefficient	Std.	p-value	95% CI	Sig%
$(0, d, 0)$	-0.0272	0.0808	0.5210	[-0.0294, -0.0250]	7.6

Note: The numbers under Coefficients, Std., and p-value are averages over 5,000 simulations, 'Sig%' reports the percentage of the estimates that are significant at 5% level.

different initial parameters. We see from Table 3 that the LLS model is rather insensitive with respect to the initial prices, initial dividend, and risk aversion parameter in terms of d. However, the changes of maximal one-period dividend decrease z_1 has a big impact. The reason seems to be that the dividend process is the driving force in the LLS model, and a change in z_1 changes the whole distribution of the dividend process. When half of the investors is endowed with \$500 and the other half endowed with \$1500, the difference in d is not significant. However, the difference in d is significant when all of the investors initial wealth is drawn from a uniform distribution on $[500, 1500]$.

Table 2. Estimates of ARFIMA$(0, d, 0)$ model

		d	Std.	p-value	95% CI	Sig%
P_0	16	-0.0286	0.0807	0.5201	[-0.0308, -0.0264]	7.7
	26	-0.0282	0.0807	0.5247	[-0.0304, -0.0260]	7.6
D_0	0.4	-0.0272	0.0808	0.5228	[-0.0294, -0.0250]	7.4
	0.6	-0.0247	0.0807	0.5193	[-0.0269, -0.0225]	6.7
α	1.45	-0.0281	0.0808	0.5172	[-0.0303, -0.0259]	7.9
	1.55	-0.0273	0.0808	0.5233	[-0.0295, -0.0251]	7.4
z_1	-0.08	0.0077	0.0837	0.6621	[0.0063, 0.0091]	0.7
	-0.06	-0.1806	0.0707	0.0848	[-0.1827, -0.1785]	70
W_0	unif.	-0.0239	0.0809	0.5241	[-0.0261, -0.0217]	6.9
	50%	-0.0279	0.0808	0.5230	[-0.0301, -0.0257]	7.7

Table 3. The t-test for the sensitivity analysis in terms of ARFIMA$(0, d, 0)$ estimates

	P_0		D_0		α		z_1		W_0	
	16	26	0.4	0.6	1.45	1.55	-0.08	-0.06	unif.	50%
t	1.167	0.840	0.047	1.935	0.760	0.125	26.696	105.53	2.546	0.554

A sensitivity analysis[1], like the one above, may be of help in determining which parameters or initial conditions, in particular, to use when one would like to calibrate (or estimate) an agent-based model using actual data. In case of

[1] The sensitivity analysis in terms of the ARFIMA $(1, d, 1)$, the probability density function, and the spectral density function yields the results more or less in line with the ARFIMA$(0, d, 0)$-case (not reported, see Li *et al.* 2006a, 2006b).

the LLS model, the dividend process seems to be an appropriate choice in a calibration exercise. By modeling it flexibly, one might become able to describe a wide range of potential distribution characteristics, so that a calibration exercise might become successful. For estimation purposes, the sensitivity analysis might shed some light on (econometric) identifiability of the parameters, with parameters having no effect on the return process not, or at most hardly, being identified.

4 Comparing Two LLS Models

Traditional theory, such as efficient market theory, casts doubts on profitability of technical trading strategies. It is interesting to study the performance of different groups of traders in the LLS setting. Zschischang and Lux (2001) investigate what will happen in the LLS model when a new type of investors is introduced, namely the constant portfolio investors, who always invest a constant proportion of their wealth in the stocks. Zschischang and Lux (2001) investigate the LLS model where initially all the investors are EMB investors (consisting of three or more subgroups). The authors found, when the market is invaded by only a small amount of constant portfolio investors (1%), that, even when these new investors are endowed with a small initial wealth and hold 1.5% of their portfolio in the stock, they eventually achieve dominance and asymptotically gain 100% of the available wealth. Here we consider a model where 0.5% of the investors are constant portfolio investors instead of RII-investors (having the same initial wealth as the other investors). These constant portfolio investors invest 1.5% of their wealth in the stock. We keep the other characteristics the same as in the benchmark model.

We performed a Wald test to investigate whether the introduction of the constant portfolio investors has a significant impact. Comparisons with the benchmark model are made in terms of log return, log price, and proportion of total wealth held by the two groups of EMB investors with different memory span. The wealth distribution is important as the influence of each type of investor depends strongly on the wealth they possess.

We consider two cases to investigate possible differences of these quantities over time. In $CaseI$, the periods under consideration are the last 100 periods, i.e., $t = 901, 902, ..., 1000$; in $CaseII$, only six time points are considered, $t = 500, 600, ..., 900, 1000$. The resulting test statistics are summarized in Table 4. In the column labeled 'Log-return', the first row reports the Wald statistics of the benchmark economy; for instance, 101.88 is the Wald statistic corresponding to the null hypothesis of equality of the average log return in the benchmark economy in periods $t = 901, 902, ..., 1000$ (with degrees of freedom between brackets), the second row reports the Wald statistics for equality of the log-returns in the new economy, and the third row reports the results of comparing the new economy with the benchmark model. To indicate the groups of EMB investors with different memory spans, we abbreviate memory span to ms. It is clear that none of the comparison statistics is significant, thus, the constant

Table 4. The comparison results with the benchmark model in terms of the mean

	Log-return	Log-price	Wealth ($ms = 5$)	Wealth ($ms = 15$)
	101.88(99)	42085.5(99)	122.20(99)	125.10(99)
$CaseI$	101.68(99)	43413.2(99)	124.84(99)	113.73(99)
	77.95(100)	76.95(100)	97.29(100)	79.13(100)
	9.53(5)	192436.7(5)	383.93(5)	772.40(5)
$CaseII$	2.65(5)	194884.1(5)	388.87(5)	757.85(5)
	8.33(6)	7.44(6)	3.37(6)	2.45(6)

portfolio investors do not cause a significant impact on the economy. We do see substantial differences across time in both economies for the price level.

Figure 1 explains why. The wealth of the constant portfolio investors decreases gradually. In the Zschischang and Lux-analysis the constant portfolio investors are the only investors who are at the opposite side of the market in case of the cycles, so that eventually they are able to gain all wealth. But in the model considered here, the RII investors for a large part take over this role by buying or selling, depending on the price being lower or higher than its fundamental value, resulting in a gradually decreasing wealth held by the constant portfolio investors.

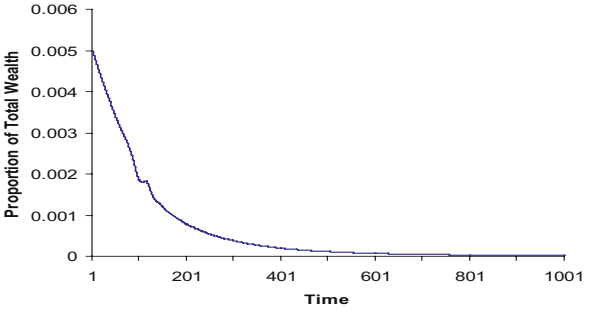


Fig. 1. Wealth held by constant portfolio investors, averaged over 5,000 simulations

5 Comparing the LLS Model with Real Life Data

From Levy *et al.* (2000), it seems that the LLS model is able to generate realistic price behaviour at a quarterly frequency. So, we use quarterly data of the S&P 500 from Datastream as representation of the real life situation, which runs from the first quarter of 1965 to the first quarter of 2003. We compare the LLS model with the real world in terms the coefficients of the ARFIMA$(0, d, 0)$ process, the probability density function, and the spectral density function. First, we estimate the ARFIMA$(0, d, 0)$ model for the S&P 500 and the results are reported in

Table 5. The parameter d is not significant, so there is no evidence of long memory in the quarterly stock return process. More important for our analysis is that the estimated d from the LLS model (the average reported in Table 1) lies within the 95% confidence interval of estimates of d from actual data, which is $(-0.1506, 0.114)$.

Table 5. Estimates of ARFIMA$(0, d, 0)$ for the S&P 500

	Coefficient	Std.	t-value	p-value
$(0, d, 0)$	-0.0183	0.0675	-0.272	0.786

Second, we consider the marginal probability density function, which can be estimated non-parametrically (see, for example, Pagan and Ullah, 1999). We apply the bootstrap to construct confidence intervals, where we follow the procedure suggested by Hall (1992). Obviously, a set of $(1-\alpha)$ point-wise confidence intervals constructed for a discretized finite interval will not achieve a $(1-\alpha)$ joint coverage probability. So, we also consider a uniform confidence band, following the suggestions by Hall (1993), see Li *et al.* (2006b) for more details.

We plot the probability density functions of the LLS model, the S&P 500, and that of a normal distribution with mean and variance equal to that of the S&P 500 in Figure 2. In addition, we also plot a 95% confidence interval in Figure 2(a) and a 95% uniform confidence band in Figure 2(b) around the S&P 500 estimates. The normal density fits within both the point-wise and uniform confidence intervals, but the LLS model based average density does not fall within the point-wise confidence interval in its entirety, while it almost fits within the uniform confidence band. Thus, the actual return distribution as a whole can be reproduced by the LLS model according to the confidence band, but the distribution seems to be shifted somewhat to the right, relative to the S&P 500 distribution.

Next, we consider the spectral density function, which is a standard way of investigating the dynamics commonly used in the economics literature (see, for instance, Diebold *et al.*, 1998). The spectral analysis yields a complete second-order comparison of the dynamic properties of model and data, providing a complete summary of the time series dynamics. We construct confidence intervals based on the procedure of bootstrapping kernel estimation proposed by Franke and Härdle (1992). We construct a uniform confidence band similar to that of Swanepoel and van Wyk (1986) but with parametric estimation, see Li *et al.* (2006b) for more details.

We present the estimated spectral density function for the S&P 500, and its corresponding 95% confidence intervals in Figure 3. In Figure 3(a) we plot the averaged spectral density function over 5,000 simulations, and in Figure 3(b) we do the same for the uniform confidence band. We find that, except for a few rather small frequencies, the LLS based spectral densities lies outside of the 95% confidence bands of the spectral density function of the S&P 500. When we compare the frequencies corresponding to the peaks of the spectra, which

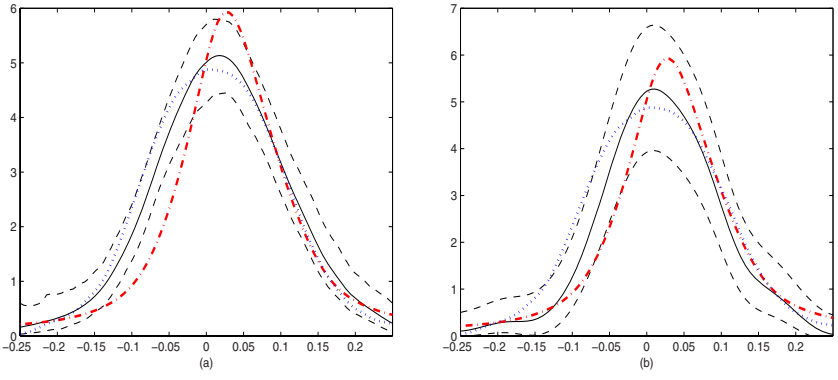


Fig. 2. The density functions of the LLS model (dash-dot line), normal distribution (dot line), and of the S&P 500 with confidence intervals (a) and confidence band (b)

describe the cycles that dominate the cyclical behaviour of the dynamics, we see that they are not at the same pace. Thus, there seems to be a large difference when we assess the second order moments between the LLS generated data and the real life data. The distribution of memory spans in the economy, however, can be expected to strongly affect the cyclical patterns in the return process and the resulting spectral density. The differences displayed in Figure 3 do suggest that substantial changes are required. We do note that matching the complete spectral density will be far from trivial for many models.

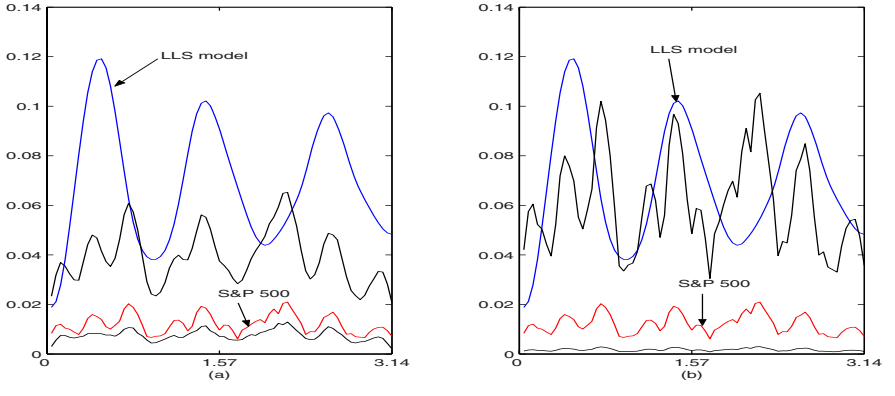


Fig. 3. The spectral density functions with its 95% confidence intervals (a) and uniform confidence band (b) of the S&P 500 and the LLS model

6 Conclusion

Econometric techniques can be used to study the agent-based models more thoroughly than the traditional eye-balling of the results, and we argue that this

should be done more often. We exploit a unique feature, our control over the number of simulations, which creates one additional dimension for the asymptotic properties of the test statistics. This allows us to ignore the estimation uncertainty present in the simulations when we compare the agent-based models with real life data. Hence, we only need to account for the estimation uncertainty of the features of the actual data. In addition, when comparing agent-based models, the control over the number of simulations provides control over the power of the tests. As an example, only 7.6% of the simulation runs resulted in a significant degree of fractional integration, while there was clear evidence of fractional integration based on all the simulation runs simultaneously.

We illustrate the use of the statistical and econometric techniques by studying one of the earlier agent-based models, the LLS model. The results from our sensitivity analysis indicate that the LLS model is robust with respect to the changes in the initial prices, the initial dividend, the risk aversion parameter, and the initial wealth. However, a change in the one period maximal decrease of dividend has a serious impact, likely, because this changes the whole distribution of the dividend process, and the dividend process is the driving force in the LLS model. This analysis also highlights the problems of estimating agent-based models, as many parameters might not affect the eventual return process and hence will not be identified.

We also illustrate how to compare the LLS generated data with the actual data. We find that while the LLS model seems to be able to provide a good description when the actual data are described by means of an ARFIMA$(0, d, 0)$ model, the LLS model is not fully in line with real data in terms of other characteristics. Taking into account the sensitivity to the dividend process, a natural way to proceed seems to be to calibrate the LLS model, focussing, in particular, on a parametrization of the dividend process.

References

Alfarano, S., Lux, T., Wagner, F.: Estimation of agent-based models: The case of an asymmetric herding model. Computational Economics 26, 19–49 (2005)

Barberis, N., Huang, M., Santos, T.: Prospect theory and asset prices. Quarterly Journal of Economics 116, 1–54 (2001)

Boswijk, H.P., Hommes, C., Manzan, S.: Behavioral heterogeneity in stock prices. Journal of Economic Dynamics and Control 31, 1938–1970 (2007)

Cochrane, J.H.: Asset pricing. Princeton University Press, Princeton (2001)

Diebold, F.X., Ohanian, L., Berkwitz, J.: Dynamic equilibrium economies: A framework for comparing models and data. Review of Economic Studies 65, 433–452 (1998)

Franke, J., Härdle, W.: On bootstrapping kernel spectral estimates. The Annals of Statistics 20, 121–145 (1992)

Hall, P.: Effect of bias estimation on converage accuracy of bootstrap confidence intervals for a probability density. The Annals of Statistics 20, 675–694 (1992)

Hall, P.: On Edgeworth expansion and bootstrap confidence bands in nonparametric curve estimation. J. R. Statist. Soc. B 55, 291–304 (1993)

He, X., Li, Y.: Heterogeneity, convergence, and autocorrelations. Quantitative Finance (to appear, 2007a)

He, X., Li, Y.: Power-law behaviour, heterogeneity, and trend chasing. Journal of Economic Dynamics and Control 31, 3396–3426 (2007b)

Hommes, C.: Heterogeneous agent models in economics and finance. In: Judd, K.L., Tesfatsion, L. (eds.) Handbook of Computational Economics, vol. 2, Elsevier Science, Amsterdam (2006)

LeBaron, B.: Agent-based computational finance. In: Judd, K.L., Tesfatsion, L. (eds.) Handbook of Computational Economics, vol. 2, Elsevier Science, Amsterdam (2006)

Levy, M., Levy, H., Solomon, S.: Microscopic simulation of financial markets. Academic Press, New York (2000)

Li, Y., Donkers, B., Melenberg, B.: Econometric analysis of microscopic simulation models. Tilburg University, CentER Discussion Papers 2006-99 (2006a), Available at: http://ssrn.com/abstract=939518

Li, Y., Donkers, B., Melenberg, B.: The nonparametric and semiparametric analysis of microscopic simulation models. Tilburg University, CentER Discussion Papers 2006-95 (2006b), Available at: http://ssrn.com/abstract=939510

Lux, T., Marchesi, M.: Scaling and criticality in a stochastic multi-agent model of financial markets. Nature 397, 498–500 (1999)

Pagan, A., Ullah, A.: Nonparametric econometrics. Cambridge University Press, Cambridge (1999)

Swanepoel, J.W.H., van Wyk, J.W.J.: The bootstrap applied to power spectral density function estimation. Biometrika 73, 135–141 (1986)

Zschischang, E., Lux, T.: Some new results on the Levy, Levy and Solomon microscopic stock market model. Physica A 291, 563–573 (2001)

Short Run Dynamics in an Artificial Futures Market with Human Subjects

Takashi Yamada, Yusuke Koyama, and Takao Terano

Department of Computational Intelligence and Systems Science,
Interdisciplinary Graduate School of Science and Engineering,
Tokyo Institute of Technology
4259 Nagatsuta-cho, Midori-ku, Yokohama, Kanagawa 226-8502, Japan
tyamada@trn.dis.titech.ac.jp, koyama@dis.titech.ac.jp,
terano@dis.titech.ac.jp

Abstract. This paper presents the computational results obtained in the strategy experiments in an artificial futures market with human subjects. Participants submit their own strategy files and they receive the performances of all the market participants in order to improve for the next round. After two-round experiments, simulations with only machine agents are run. We find that the time series data support so-called stylized facts in some regards and that experiments of human subjects seem to make the prices be closer to a theoretical value.

1 Introduction

The advent and development of computer technology have improved the methodology of experimental and cognitive economics. Before computer era researchers implemented so-called "pencil-and-paper" experiments and thus the amounts of results obtained had to be somewhat limited. In the present era, on the other hand, higher computer abilities enable us to implement iterated experiments by changing setup conditions and thereby to collect accurate data for proper analyses (a good review is made by Duffy [2] and an example of computer-oriented cognitive economics is by Ueda et al. [12]).

Such an improvement has led to create a bland-new economics: agent-based computational economics (e.g. Tesfatsion and Judd [11]). This approach has been able to explain micro-macro relations in economic dynamics and to cover the fields of which experimental economics with human subjects is difficult to implement such as long run or extensive simulation.

In recent years there have been several collaborations between experimental/cognitive economics and agent-based computational economics. One of the attempts is to incorporate the findings of experiments into the frameworks of agent modelling and vice versa (also Duffy presents a good explanation [2]). The other is to develop a methodology or toolkit in order to make use of teaching computational economics (e.g. Kendrick et al. [6]). At the same time, practitioners have made use of computer abilities in order to trade in the markets, analyze

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1092–1101, 2007.

financial data, and train traders (e.g. Ueda et al. [12]). The same can be said to personal investors. For instance, they try to experience and understand market mechanisms through trading artificial economies (e.g. LIVE by Simplex Institute, Inc. [8] and SimStockExchange by Hoffman et al. [9]). Or, hasty investors may take advantage of trading agent services as program trading, namely such market participants need to write some programs to express their own trading strategies. Hence the diffusion of computer technology and changes in trading environments lead both investors and financial institutes to take into account utilization of information technology for investing service. In other words, agent-based computational finance is also considered as a kind of service so long as program trader exists in the markets.

"UnReal Market as an Artificial Research Testbed" (hereafter U-Mart) is one of the common used toolkits in experimental and agent-based economic studies [13]. This is an artificial futures market in which human subjects and trading agents take part in at the same time. By using this testbed, researchers try to clarify the market dynamics/mechanism, to make use of this tool for engineering and economic studies, and to do a campaign for enlightenment. This paper explores what U-Mart can contribute to computer oriented economics and what it should be required for the future through the simulation results with and without human subjects.

The rest of this paper is organized as follows: The next section describes the experimental setup. Section 3 presents the results of time series analyses using sample paths generated. Section 4 discusses future perspectives of computational economics and its usage for service sciences. And finally, Section 5 concludes this paper.

2 Experimental Design

The experiment was implemented as a part of a course "System Modeling," an engineering introduction to computational intelligence and systems science in the graduate school of science and engineering program at Tokyo Institute of Technology. Participation was a course requirement for master's course students. Almost all the students had no prior knowledge about financial markets, but several students had some skills in computer programming. Note that the procedure employed in this study happened to be similar to that in Hommes et al. and Sonnemans et al. [4,10] and that this course does not intend to teach how to make more money in financial markets.

2.1 Tutorial

The objectives of this tutorial were to provide the students with some experiences with operating U-Mart and to give lectures about computer programming. After installing U-Mart for each personal computer, three introductory sessions were held as follows: In the first session, a trading pre-contest was implemented. In this session, only human subjects took part in the artificial market in order to grasp how a futures market ran. In the second and third sessions, computer

programming lectures were given. While the students were taught elementary JAVA programming in the first half of the classes, they learned how to create a machine agent using a template file distributed in the second half of the lecture.

2.2 Strategy Experiment

The experiments lasted two weeks, each of which had one round. In each round, subjects had to submit a strategy file in JAVA. Students could submit their own strategy anytime before the previous day of the contest. In the first round subjects had about two weeks to create agents, while in the second round they had only one week to revise their strategy. In other words, they could make machine agents after taking all the introductory lectures. The number of submissions were 87 and 89 of 89 registrations respectively. The instructors and two teaching assistants checked these strategies for not having any bug or error. As a result, two strategies were excluded in the first round, and three were in the second.

In each round we implemented an experimental asset market with human subjects and submitted strategies only one time and a computer simulation with only machine agents 10 times. The reason why we could not conduct iterated experiments in case of the market with students is human subjects surely learn from the past events. The two kinds of time series spot data, the one is NIKKEI225 and the other is USD/JPY, were converted such that the mean and the variance were all equal to those of originally installed data, J30. Since each simulation run had 20 days each of which had eight bid/offer matching done on a board, one matching could be considered as one-hour long. Moreover, the human subjects had about 20 seconds in each matching for their decision makings. Market participants were allowed to do infinitely short-selling so long as their budget permitted, but the ones who had gone bankrupt could not take part in the market anymore (other setups are described in Table 1). At the end of each round, the subjects received open information about all the source codes, order information, historical data (price and volume), and the rankings of the strategies and human subjects by final wealth. After experiment students revised their strategy based on the results and submitted for the next competition (even if the third round did not take place).

Problems often addressed by many researchers are motivations of subjects and attempts to obfuscate the market. The former problems would be overcome by letting the participants be financially motivated, namely instructors announced that the most profitable human subject and the student who created the winner agent could receive sweet treats for the amount of 10 dollar. On the other hand, with respect to the latter obstacle we did not prohibited them from making a destabilizing machine agent since we knew that such an attempt would be quite hard to succeed due to the existence of nearly 200 market participants plus originally installed machine agents [1] as Hommes et al. have pointed out [4]. Fortunately, all the strategies submitted which will be explained in the sequel were ordinal.

[1] They are as follows: one trend follower, one contrarian, two random walkers, two RSI traders, two moving average strategies, one arbitrager (he/she focuses on the spread between spot price and futures price), and one stop loss trader.

Table 1. Experimental setups

Item	Memo
Initial wealth	One-billion
Initial holdings	No
Ordering for human subjects	Limit order and market order
Ordering for machine agents	Limit order only
Cancellation of orders	Allowed only for human subjects
Risk free rate	0.1
Trading unit	1000-fold
Commission	300-thousand per unit
Credit taking	Up to 30-million

Table 2. Characteristics of submitted strategies (Some strategies have more than two characteristics)

	First round	Second Round
Random	5	2
Stop loss	10	11
Trend follower	20	20
Contrarian	4	5
Moving average	22	20
Spot-futures spread	28	31
Others	8	10
Total strategies	87	89

3 Results

3.1 How Are Agents Created?

In agent-based computational finance models, the characters of agents are mostly bounded rational, namely the characters of agents are usually fundamentalists, chartists, deterministic, or ones using evolutionary algorithm. Before presenting the results of market dynamics, we will briefly review general distinctions of submitted strategies.

Table 2 shows main characteristics of the strategy files. About one-thirds are arbitragers, namely they think that the futures price will eventually converges to the spot price. The rest strategies are something like Markov-property or moving average ones. That is to say, the former strategies can be considered as ones with characters of fundamentalists and the latter ones are chartists. Around 10 strategies employ stop loss orders, which is because the U-Mart allows market participants to do more than two orders at a time. Finally, around 10 other strategies are more complex ones, namely they consist of neural-network program, classifier systems, or reinforcement learning.

3.2 Stylized Facts

Financial market data contain many statistical properties called "stylized facts" for which traditional economics is difficult to explain. Some of them are about price movements per se and others are the relations between trading volumes, and price movements or volatility. We will focus on the following four properties which seem to be the most popular and significant facts and have been reproduced by several agent-based simulation models (e.g. Hommes, and Lux and Marchesi [3,7] for example)[2]:

- Exchange rates and stock prices have almost unit roots.
  To check if a time series has a unit root, one often employs three unit root test, Dickey and Fuller test, Augmented Dickey Fuller test or Phillips-Perron test. If the p-value is less than a threshold value, 0.05 for instance, then the series has a unit root.
- Returns have fat-tailed distributions.
  Fat-tailed distribution is whose density function decreases in a power-law fashion. But according to Lux and Marchesi [7], the fact is seen for returns at weekly or shorter time scale.
- Returns per se cannot be predicted, namely they have almost zero autocorrelations.
- Return distribution shows long memory, namely absolute or squared returns are significantly positive and decrease slowly as a function of the lags.

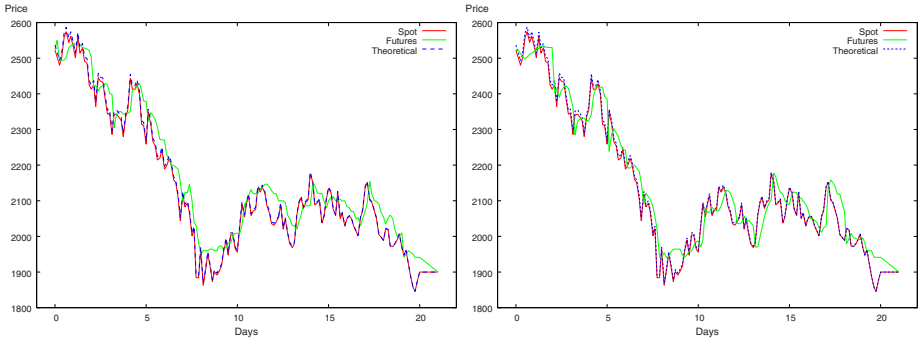
3.3 Market Dynamics

This part of the section reports the results of time series analyses using one laboratory experiments and one of 10 simulation runs for each round. Each time series plot is presented in Figure 1. The realized futures prices seemed to trace the spot prices, but sometimes large jumps are observed because human subjects did large amount of market order. In other words, the differences between simulated prices and spot ones with only machine agents are smaller than those in strategy experiments.

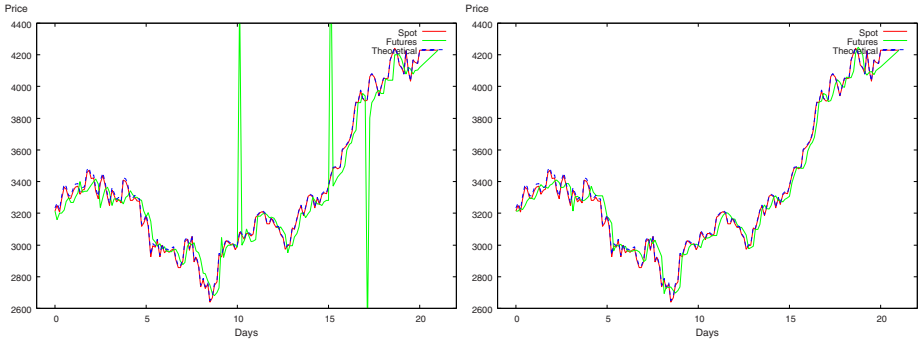
First, Table 3 depicts the p-values of three unit root tests, DF test, ADF test, and PP test for generated price series. Those tests prove that no simulation model or setup except one rejected the null hypothesis of the presence of a unit root. One possible explanation for non-rejected property is there were some jump processes in the second laboratory experiment (Figure 1b).

Second, Figure 2 presents normal probability plots of simulated return series. Due to the smallest observations for daily data, only hourly data are shown. The reason why the curves indicate fat-tailed distribution is when one takes up time series data at shorter time scale, a jump process is likely to take place. Therefore

[2] With respect to the relations between price changes and trading volumes, there is a good and classical review by Karpoff [5], and Chen and Liao have clarified the mechanism by agent-based approach focusing on Granger's causality [1]. But this item will be omitted due to the smaller number of observations.



a. First round



b. Second round

Fig. 1. Time series plot (left panel: experiments with human subjects and machine agents, right panel: with machine agents only, horizontal axis: term, and vertical axis: price)

in this regard trading strategies with little experiments led market dynamics to the ones similar to real markets.

Third, Figure 3 reports auto-correlation functions which test for long memory distinctions of time series data. Clearly the generated returns do not show such a distinction, namely there is unrealistic pattern. Though we did not conduct deeper analyses, the possible reason may be that the time horizons of moving average strategies were similar to each other. But this is still an open question.

Finally, Figure 4 and Table 4 show whether generated sample paths converged to a theoretical value which is derived from

$$F_t = S_t \cdot (1 + r_t \cdot \tau_t / 365) - d_t$$

where F_t, S_t, d_t, r_t, and τ_t are futures price, spot price, dividend, risk free rate, and days to maturity respectively. Since we postulated that there was no dividends paid to shareholders, the second term of the equation above is set to zero. Both the exhibits reveal that even though the sample paths in the first round had no jump processes the time series were hard to converge to the fundamental value. This fact is also supported by a positive Lyapunov exponent.

Table 3. Unit root tests (p-value)

			DF	ADF	PP
First round	(w. human subjects)	Daily	0.677	0.382	0.795
		Hourly	0.742	0.785	0.817
	(w/o human subjects)	Daily	0.710	0.448	0.833
		Hourly	0.687	0.635	0.748
Second round	(w. human subjects)	Daily	0.934	0.911	0.965
		Hourly	0.010	0.046	0.010
	(w/o human subjects)	Daily	0.927	0.909	0.963
		Hourly	0.969	0.950	0.973

DF: Dickey and Fuller test
ADF: Advanced Dickey and Fuller test
PP: Phillips and Perron test
If the p-value is less than 0.05, then a series has a unit root.

Besides, this distinction is observed for all the 10 sample paths in the U-Mart with only machine agents. For one thing, the subjects did not get accustomed to JAVA programming and the mechanism of financial markets. Consequently, the prices formed in the market with simple but random-like traders became to be more chaotic. On the other hand, time series data with a negative Lyapunov exponent in the second round appeared to be more stable in spite that the laboratory experiment had a few large jump.

4 Discussion

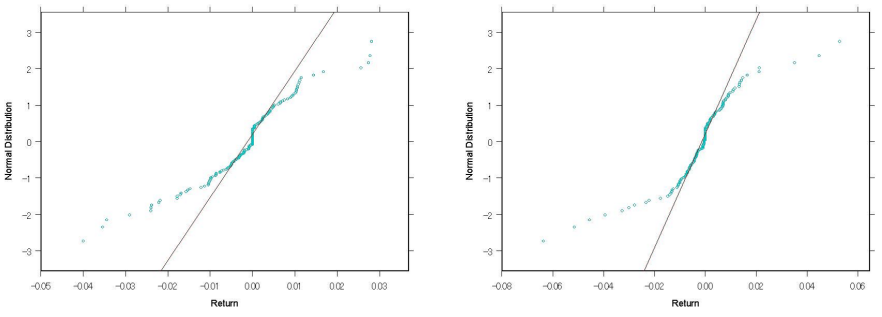
4.1 What Is Done, What Is Not?

On the one hand, one preferable result is that the prices would be closer to the theoretical value when subjects had some knowledge of the market and experiments in spite that the market is not the same as that in Hommes et al. [4]. Moreover, unit root property and fat-tailed distribution are observed when one takes up hourly time scale. This supports the fact that the market participants are boundedly rational and heterogeneous.

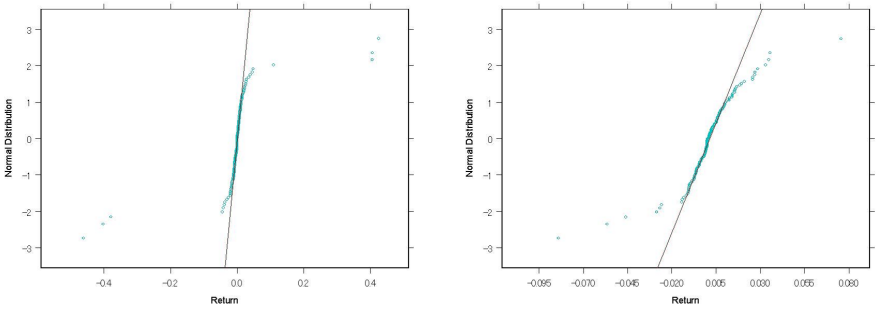
On the other hand, long run dynamics could not be analyzed in this setup at all because of time restrictions or computer/network problems. Therefore auto-correlation functions, BDS statistics, and relations between price changes and trading volumes were omitted. Besides we did not compare the results with/without human subjects and examine what if we add procedures of risk management, order cancellation, or market order to the template file.

4.2 Future of U-Mart as Service Sciences

It has been about a decade since the birth of U-Mart and lots of contributions have been made in economic and engineering literature. At the same time, it has been widely used in educational program for teaching computational economics.



a. First round



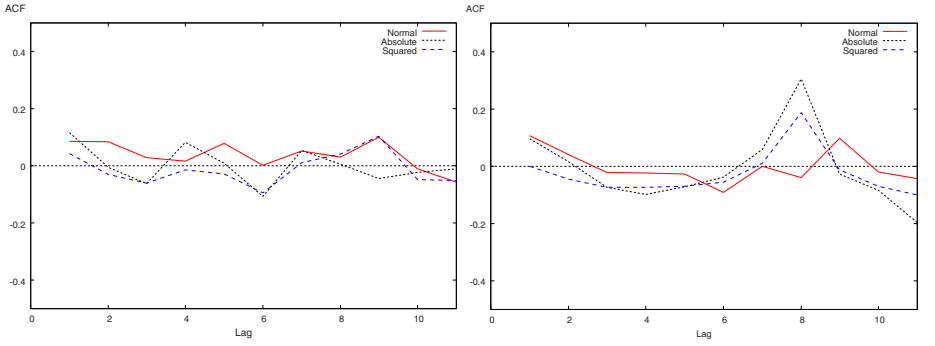
b. Second round

Fig. 2. Normal probability plots (left panel: with human subjects, right panel: machine agents only, horizontal axis: return, vertical axis: cumulative distribution)

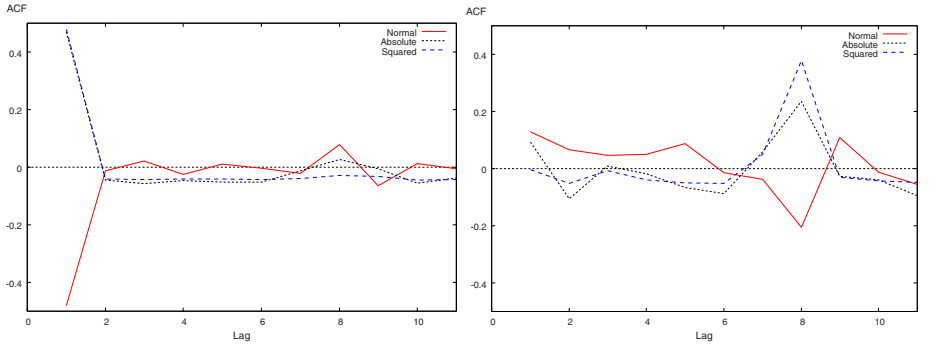
In order to keep these trends, we believe that the following points should be grappled with in the near future: Firstly, more efforts to help researchers and practitioners understand the mechanism of markets and behaviors of market participants should be done. As far there are several studies about risk control abilities of human subjects, but no research about combining such findings into the behavioral economic theory is found. Secondly, for engineering program, instructors need to let students be interested in what social science is all about as well as writing a more complex/sophisticated machine agent. This is because being conversant with social science for engineering students as well as having skills in computer programming for economic students is required for understanding of computational economics.

5 Conclusion

This paper reports the strategy experiments in an artificial futures market with human subjects in order to verify how current agent-based computational finance is useful for service sciences. Two rounds experiment and simulation re-

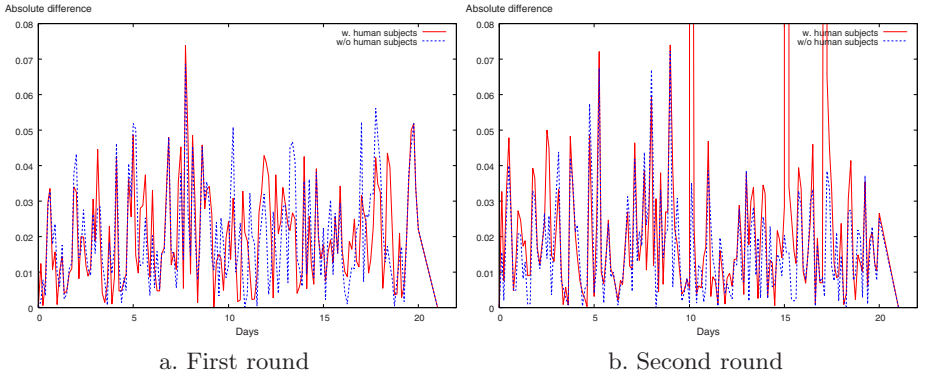


a. First round



b. Second round

Fig. 3. Auto-correlation functions (left panel: with human subjects, right panel: machine agents only, horizontal axis: lag, vertical axis: ACF)



a. First round b. Second round

Fig. 4. Absolute difference from the theoretical value (horizontal axis: term, vertical axis: difference between simulated futures price and theoretical one)

Table 4. Number of terms within a 0.01 range from a theoretical value

	Daily (20 obs.)	Hourly (160 obs.)
First round (w. human subjects)	3	46
First round (w/o human subjects)	1	43
Second round (w. human subjects)	5	52
First round (w/o human subjects)	6	64

sults afterward reveal that an appropriate education program and some experiments of human subjects could make market dynamics the ones observed in real markets, namely more experienced machine agents and trading behaviors made a chaotic dynamics disappear even if the experiments were implemented under constrained environments. Instead, we also confirm that analyses of long run dynamics and the collaboration between establishment of course curriculum and experiments are required for future of computational economics and service sciences.

References

1. Chen, S.-H., Liao, C.-C.: Agent-based computational modeling of the stock price-volume relation. Info. Sci. 170, 75–100 (2005)
2. Duffy, J.: Agent-based models and human subject experiments. In: Tesfatsion, L., Judd, K.L. (eds.) Handbook of computational economics: agent-based computational economics, vol. 2, pp. 949–1012 (2006)
3. Hommes, C.H.: Financial markets as nonlinear adaptive evolutionary systems. Quant. Finan. 1, 149–167 (2001)
4. Hommes, C.H., Sonnemans, J., Tuinstra, J., van de Velden, H.: A strategy experiment in dynamic asset pricing. J. Econ. Dyn. Control 29, 823–843 (2005)
5. Karpoff, J.M.: The relation between price changes and trading volume: a survey. J. Finan. Quant. Analysis 22, 109–126 (1987)
6. Kendrick, D.A., Mercado, P.R., Amman, H.M.: Computational economics: help for the underestimated undergraduate. Comput. Econ. 27, 261–271 (2006)
7. Lux, T., Marchesi, M.: Volatility clustering in financial markets: a microsimulation of interacting agents. Int. J. Th. Appl. Finan. 3, 675–702 (2000)
8. Simplex Institute, Inc., http://www.simplexinst.com/english/products/index.htm
9. SimStockExchange: http://www.simstockexchange.com/
10. Sonnemans, J., Hommes, C.H., Tuinstra, J., van de Velden, H.: The instability of a heterogeneous cobweb economy: a strategy experiment on expectation formation: J. Econ. Behav. Org. 54, 453–481 (2004)
11. Tesfatsion, L., Judd, K.L. (eds.): Handbook of computational economics: agent-based computational economics, vol. 2. North-Holland, Amsterdam (2006)
12. Ueda, K., Uchida, Y., Izumi, K., Ito, Y.: How do expert dealers make profits and reduce the risk of loss in a foreign exchange market? In: Proc. of the 26th annual conf. of the Cognitive Science Society, Chicago, USA, pp. 1357–1362 (2004)
13. U-Mart project: http://www.u-mart.org/

Video-Based Conjoint Analysis and Agent Based Simulation for Estimating Customer's Behavior

Hiroshi Sato, Masao Kubo, and Akira Namatame

Dept. of Computer Science, National Derense Academy of Japan,
1-10-20 Hashirimizu, Yokosuka, Kanagawa 238-8686, Japan
{hsato, masaok, nama}@nda.ac.jp

Abstract. Conjoint analysis is a statistical technique to reveal customers' invisible preference using series of questions regarding tradeoffs in products. In this paper, we propose a new variant of this technique that uses products layout and customers' actions in a store instead of conjoint cards and answers. We demonstrate the effectiveness of this method by making agent-based in-store simulator that can reproduce the congestion in a store. The parameters of the agents in the simulator were determined by our technique – video-based conjoint analysis.

Keywords: Service, Science, Conjoint Analysis, Random Utility Maximization Model, Multinominal Logit Model, Behavior Analysis, Agent-based simulation.

1 Introduction

Service science is an attempt to treat service as a problem of science and engineering. Progress of the processing speed of today's computers and the methodology of software designing make this approach possible [1]. Agent Based Simulation (ABS) is very powerful tools for service science. Concept of ABS is quite different from traditional numerical calculation of differential equations. ABS is effective especially in social science because bottom-up approach of ABS have high affinity with constructing complex systems [2].

In order to make a valid ABS, design of the agent is the most important thing. The term "agent" is used as the meaning of "customer" throughout this paper. To make a reliable agent, we must know customers' preference correctly, but at the same time, this is the most difficult thing in service industry.

There are two directions regarding customer survey: (1) asking to customers and (2) observing customers indirectly. Questionnaire [3] is the typical example of the former and behavior analysis [4] is the typical example of the latter. Both have pros and cons. Questionnaire is simple and convenient, but it usually needs incentive such as money to have people answer the questionnaire, and what is worse that people can not answer correctly to the questionnaire. Behavior analysis is based on the observation of the agents (animal or human). It can get more precise result because

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1102–1111, 2007.

the action of agents represents what they think. But this method needs special instruments for recording or analyzing.

Conjoint analysis is a variation of questionnaire [5][6]. Through the series of indirect questions regarding tradeoffs of products, it can reveal the relative strength of the attributes of the products for customers. This method can prevent examinee to deceive examiner because the question is indirect and it is difficult to estimate what the examiner want to know.

Conjoint card is special tool of this method. The cards are examples of products that have tradeoff among many attributes such as price, size, or performance. Examinees are required to sort or choice these cards by their preference. The results of sorting or choice can be used to estimate relative importance of the attribute to the examinees.

By the way, the actions of customers in a store, such as turning the corner to some direction or stopping at the front of some shelf, can be thought as choice or sort. POS system is used for long time to analyze customers, but it stores only purchase data. Given the recent and rapid development of IT tools, it is relatively possible to store, retrieve and analyze almost every action of customers. In this paper, we propose an extension of conjoint analysis that can be carried out without conjoint card. In this method, each actual product in the store represents each conjoint card, and the stored records of customers' actions are translated into the choice or sort of the conjoint card.

The rest of this paper consists of the following sections. In Section 2, conjoint analysis – the theoretical background of this study – is reviewed. In Section 3, we propose a new method that extends conjoint analysis that doesn't need conjoint card. The effectiveness of the method is demonstrated in Section 4. We discuss the possibility of creating ABS using this result in Section 5. Section 6 is about conclusion and future work.

2 Conjoint Analysis

2.1 Basic Concept

Conjoint analysis is the statistical technique used in marketing research to determine the relative strength of the attributes of the product or service. It originated in mathematical psychology [5] [6].

The following is the typical procedure of conjoint analysis: A product or service is described in terms of a number of attributes. For example, PC may have attributes of size, mobility, memory, hard drive disk, CPU, and so on (Fig. 1). Each attribute can be broken into a number of levels. Examinees would be shown a set of products created from combination of levels and asked to choice, sort, or rate bye their preferences. Using the regression analysis, the implicit utilities for the levels can be calculated.

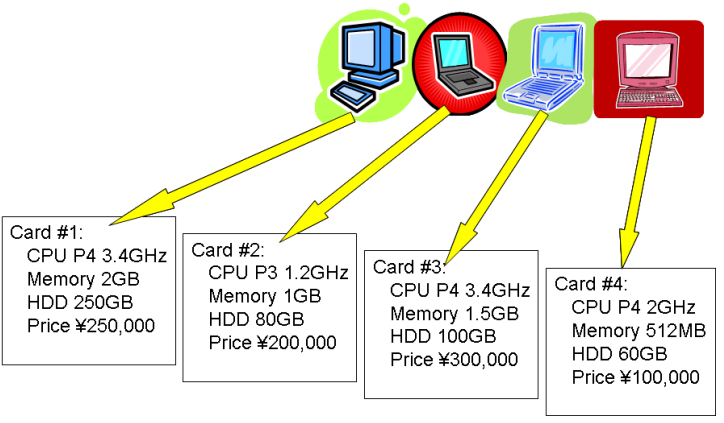


Fig. 1. An example of conjoint card: The combination of attributes of PC makes conjoint card. Examinees should asked to choice or sort these cards by their preferences.

2.2 Theoretical Background

To make the utility function, we use Random Utility Maximization (RUM) Model [5]. In RUM, the utility is formulated in the shape of linear functions (Eq. (1)).

$$U_{in} = \beta_1 x_{1in} + \beta_2 x_{2in} + \cdots + \beta_k x_{kin} + \varepsilon_{in} = V_{in} + \varepsilon_{in}, \tag{1}$$

where,

U_{in} : Utility of i - th choice of n - th people,
x_{kin} : k - th explaining variable of i - th choise of n - th people,
β_k : k - th parameter.

The choice probability can be calculated by Eq. (2).

$$P_n(i) = \Pr[U_{in} > U_{jn}, \quad \text{for } \forall j, \ \ i \neq j] \tag{2}$$

If the number of the choice is two, we can rewrite Eq. (2) as Eq. (3)

$$\begin{aligned} P_n(i) &= \Pr[U_{in} \geq U_{jn}] \\ &= \Pr[V_{in} + \varepsilon_{in} \geq V_{jn} + \varepsilon_{jn}] \\ &= \Pr[\varepsilon_{jn} - \varepsilon_{in} \leq V_{in} - V_{jn}] \\ &= \Pr[\varepsilon_n \leq V_{in} - V_{jn}] \\ &= \mathrm{CDF}_{\varepsilon}(V_{in} - V_{jn}) \end{aligned} \tag{3}$$

If the CDF is logistic distribution, Eq. (3) will be Eq. (4).

$$P_n(i) = \frac{\exp(\mu V_{in})}{\sum_{j=1}^{J} \exp(\mu V_{jn})}, \quad i = 1, \ldots, J \tag{4}$$

Parameter estimation is done by Maximum Likelihood Estimation (MLE). Log likelihood is often used for actual calculation (Eq. (5)).

$$L = \prod_{n=1}^{N} \prod_{i=1}^{J} P_n(i)^{d_{in}} \text{ , or } \ln L = \prod_{n=1}^{N} \prod_{i=1}^{J} d_{in} P_n(i) \tag{5}$$

where,

$$d_{in} = 1 \text{ (if n - th person selects i - th choice),}$$
$$= 0 \text{ (if not)}$$

3 Video-Based Conjoint Analysis: Conjoint Analysis Without Conjoint Cards

In this section, we propose the extension of conjoint analysis. Usually, we need conjoint card to carry out conjoint analysis. As we mentioned in Section 2, conjoint analysis consists of the following two parts: (1) making conjoint cards, and (2) asking examinees to sort or choice the cards. We can substitute each part using IT tools. We discuss the possibility of extension of these two parts in order.

3.1 Making Conjoint Card from Store Shelf

We have to remember the each conjoint card represents a possible product which has some combination of attributes. Fortunately, there are many kinds of possible products on a shelf in usual convenience store. Then we can approximately think the products on the shelf as the display of conjoint cards and the action taken at the front of the shelf can be translated into the action against conjoint cards.

Table 1. Attributes and its levels of the food and drink sold in convenience store

Attributes	Level
Category	Food, Snack, Drink
Temperature	Hot, Normal, Drink
Life time	Short, Long
Price	High,, Normal, Low

In particular, if the products in a convenience store are intended, food and drink will be the object of conjoint analysis. Table 1 shows the example of possible attributes and level of food and drink. Fig. 2 and Fig. 3 show the shelf layout in some convenience store and the translation of products on the shelf to conjoint card.

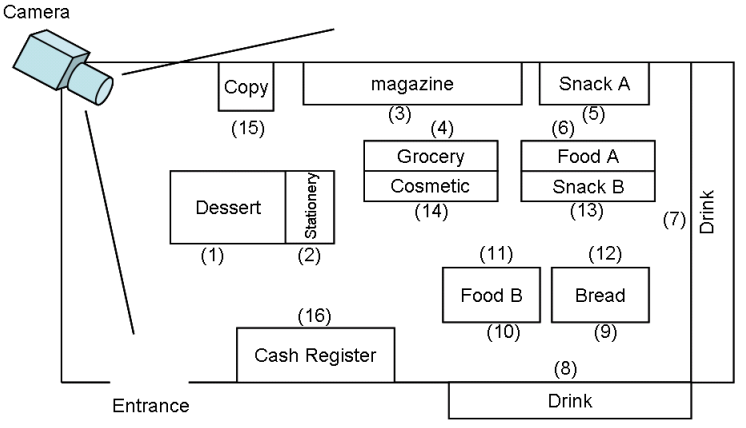


Fig. 2. The layout of a convenience store: Each shelf in the store is classified by the products

```
Card #1 (Food, Hot, Long, Low) --> Position (6)
Card #2 (Food, Hot, Shot, High) --> Position (11)
Card #3 (Food, Normal, Short, Normal) --> Position (12)
Card #4 (Snack, Cold, Short, Normal) --> Position (1)
Card #5 (Snack, Normal, Long, Low) --> Position (5)
Card #6 (Snack, Normal, Long, High) --> Position (13)
Card #7 (Drink, Cold, Long, Normal) --> Position (7)
Card #8 (Drink, Cold, Short, Normal) --> Position (8)
```

Fig. 3. An example of making conjoint card: the product on the shelf can be translated into the conjoint cards. "Position" represents the shelf number in Fig. 2.

3.2 Reading Mind from Customers' Action Instead of Asking Questions

The above idea is maybe considered at the time in the past, but there is no means to capture the customers' action at the time. However, recent development of IT tools allows us to monitor and record customers every action. For example, RFID or very smart image processing methods can be used for this purpose.

We select spending time at the front of each shelf in the store as the substitution of customers' choice for conjoint analysis. Longer spending time can represents they are interested in the products and vice versa. POS data is unsuitable for this purpose because it only tells us whether it sold or not. We need more detailed data such can represent how they want or how they wander to buy some products.

In this study, we use the video-based IT tools because it is easy to check the log (just watch the video). We adopt Vitracom's SiteView [7] for the analysis of video image. It can count, detect and track objects crossing the counting line in both direction at very high density. Fig. 4 shows the screenshot of counting people.

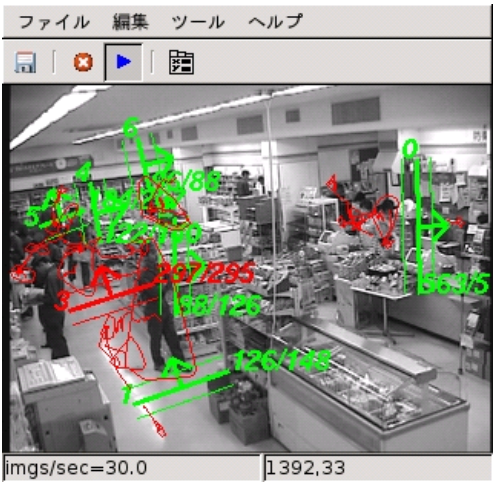


Fig. 4. Analyzing customers' action: Vitracom Siteview is used to analysis of customers' behavior. Siteview is video-based counting device. It can count objects crossing the counting line and can detect and track the objects using optimization algorithms.

4 Experiments

In order to examine the effectiveness of our method, experiments were done in two situations at the convenience store located in our campus. Table 2 shows the condition of the experiments. Fig. 5 shows scenes of these two situations.

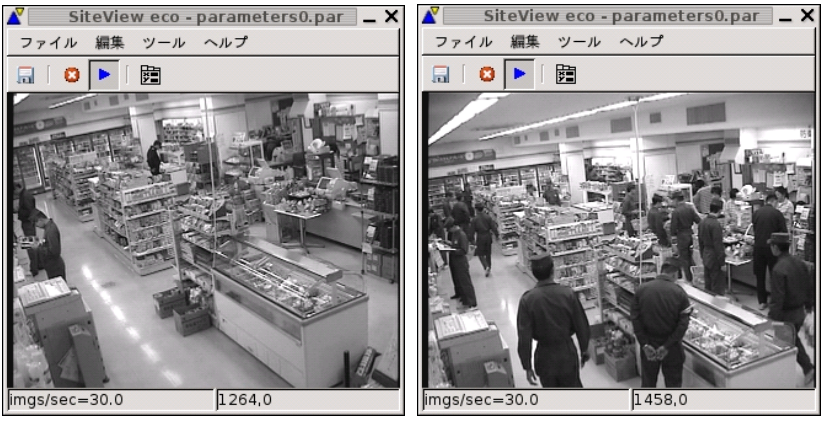


Fig. 5. Typical scene of Situations – (Left) Before meal: 16:00-17:00. In this situation, the store is sparsely populated (2 ~ 3 people in the store). (Right) After meal: 18:00-19:00. In this situation, the store is densely populated (10 ~ 15 people in the store).

Table 2. Condition of the experiments

Date	March 16th, 2007
Location	Convenience store in our campus (National Defense Academy of Japan)
Time	16:00-17:00 (Situation 1) 18:00-19:00 (Situation 2)
Objects	Food and drink

4.1 The Results of Experiments

SPSS conjoint [8] is used to carry out these investigations. Fig. 6 shows the relative importance of attributes of the products in both experiments. Relative importance among attributes is almost same in situation 1 (before meal), but the category is the most important attributes in situation 2 (after meal).

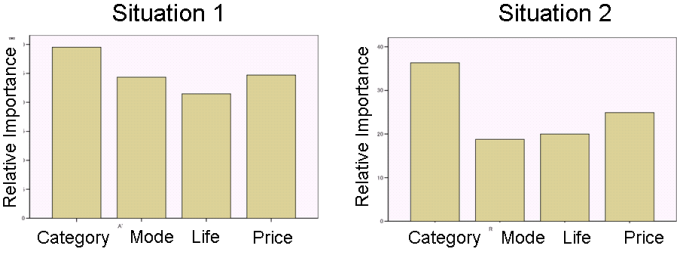


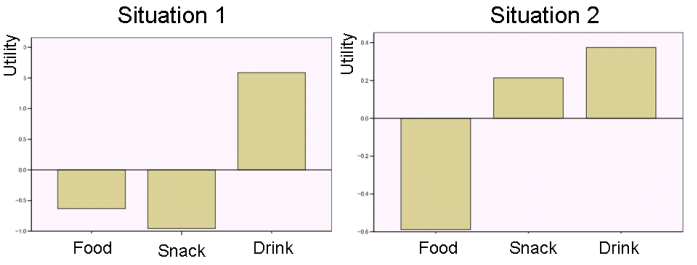
Fig. 6. The difference of relative importance to the utility function

Fig. 7 (a)-(d) shows the effects of each attributes on the utility functions of the customers. These results clearly show the change of the parameters of the utility functions.

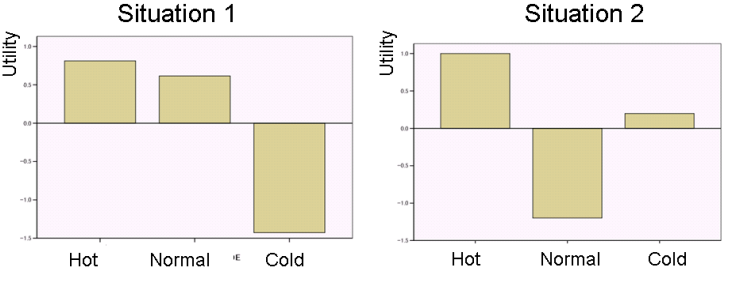
4.2 Simulating the Efficiency of the Possible Products

When we get the estimation of the parameters of utility function for customers, we can simulate the rating of nonexistent products by calculating the utility function. The following two products show the opposite utilities between the situations.

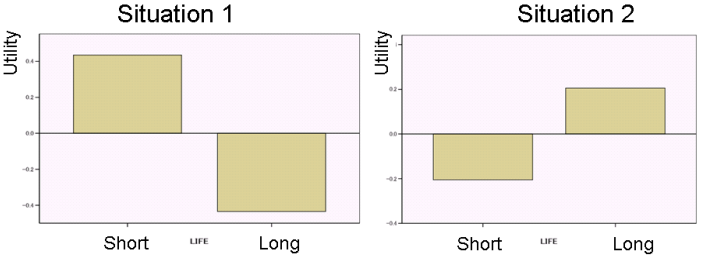
- (Drink, Normal, Short, Low) gets high rating in situation 1 (= 5.61), but low rating in situation 2 (= 1.24).
- (Snack, Hot, Short, Middle) gets low rating in situation 1 (= 2.22), but high rating in situation 2 (= 4.00).



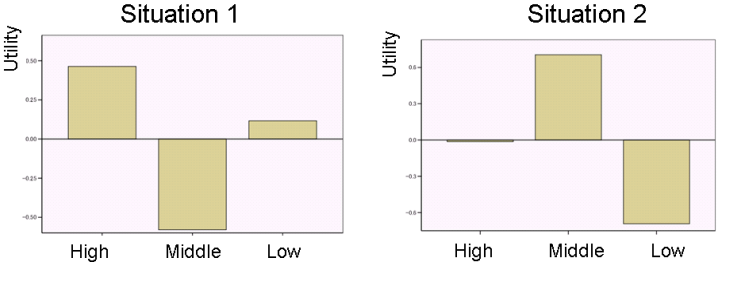
(a) Category of the Food



(b) Temperature of the food



(c) Lifetime of the food



(d) Price of the food

Fig. 7. The effects of each attribute on the utility function of customers

5 Agent Based Store Simulator Incorporating the Results

We now are able to create agent based store simulator. We adopted AnyLogic [9] Pedestrian Library for this simulator. This library enables to make natural flow of people from point to point with route finding and collision avoidance mechanism.

This simulator has the same layout of the store is as shown in Fig. 2. Agents move around according to the sequence of waypoints in a store. The place in front of each shelf corresponds to each waypoint. Agents enter from the entrance and visit few waypoints and go outside from the exit. Waypoints are selected stochastically based on the preference of each agent. The average number of the selected waypoints for each agent is three. This number is set from the observation.

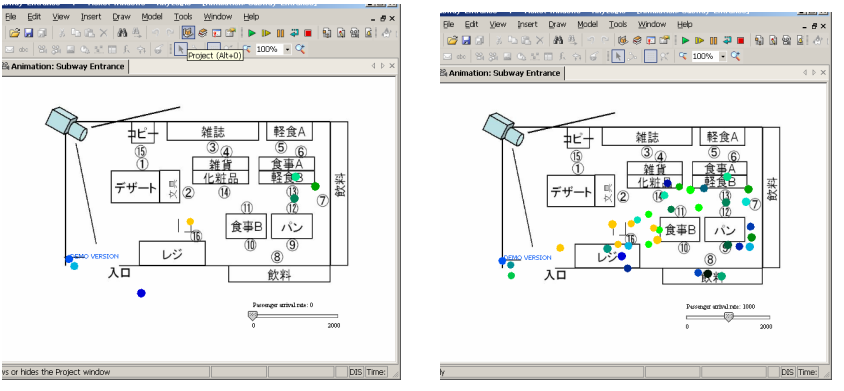


Fig. 8. Store simulator using AnyLogic Pedestrian Library: (Left) Off hours, (Right) Rush hours

From the right part in Fig. 8, congestion can be seen at the front of cold drink shelf when the number of the people in the store increases. This result corresponds with real data that we observed. The next step is to find better layouts that can prevent congestion and this is an ongoing project.

6 Conclusion

Knowing customers' preference is the most important but difficult thing in marketing. We propose new investigating method that combines questionnaire and behavior analysis. In this method, customers are modeled as agents that maximize their utilities. The parameter of utility function of the agent is estimated with their actions in store such as flow line and sojourn time. More precisely, agents' action is used for creating the answer to the conjoint cards that consist of questions regarding tradeoff of products.

Experiments done in some convenience store show this method can differentiate the change of agent's preference. In the experiments, we can obtain good estimate value because of the several particularity of the situation. For example, (1) everyone

is regular customer and familiar with the layout, (2) the target products (food and drink) have obvious tradeoff. We need to develop this method for broader range of situation.

This method can simulate not only existent product, but also nonexistent products. We tried to reflect this result in building customer-model for agent-based store simulator and reproduced the congestion that corresponds with observed data. As a future work, more precise design of the simulator and finding better layout that can prevent congestion is planned.

References

1. Spohrer, J., Maglio, P., Bailey, J., Gruhl, D.: Steps Toward a Science of Service Systems. Computer 40(1), 71–77 (2007)
2. Epstein, J.: Generative Social Science: Studies in Agent-Based Computational Modeling. Princeton University Press (2007)
3. Hague, P.: Questionnaire Design, Kogan Page (1993)
4. Malott, R.W., Malott, M.E., Trojan, E.A.: Elementary Principles of Behavior, Prentice Hall College Div. (1999)
5. Green, P., Srinivasan, V.: Conjoint analysis in consumer research: Issues and outlook. Journal of Consumer Research 5, 103–123 (1978)
6. Gustafsson, A., Herrmann, A., Huber, F. (eds.): Conjoint Measurement: Methods and Applications. Springer, Heidelberg (2006)
7. Vitracom Siteview Web Page: http://www.vitracom.de
8. SPSS Web Page: http://www.spss.com
9. AnyLogic Web page: http://www.xjtek.com/

Effect of the Number of Users and Bias of Users' Preference on Recommender Systems

Akihiro Yamashita[1], Hidenori Kawamura[1],
Hiroyuki Iizuka[2], and Azuma Ohuchi[1]

[1] Graduate School of Information Science and Technology, Hokkaido University
North 14, West 9, Kita-ku, Sapporo, Hokkaido 060-0814, Japan
{yama,kawamura,ohuchi}@complex.eng.hokudai.ac.jp
http://harmo.complex.eng.hokudai.ac.jp/
[2] Department of Media Architecture, Future University-Hakodate
116-2 Kamedanakano-cho, Hakodate, Hokkaido, 041-8655, Japan
ezca@fun.ac.jp

Abstract. Recommender System provides certain products adapted to a target user, from a large number of products. One of the most successful recommendation algorithms is Collaborative Filtering, and it is used in many websites. However, the recommendation result is influenced by community characteristics such as the number of users and bias of users' preference, because the system uses ratings of products by the users at the recommendation.

In this paper, we evaluate an effect of community characteristics on recommender system, using multi-agent based simulation. The results show that a certain number of ratings are necessary to effective recommendation based on collaborative filtering. Moreover, the results also indicate that the number of necessary ratings for recommendation depends on the number of users and bias of the users' preference.

Keywords: Recommender System, Collaborative Filtering, users' preference, Multi-agent Simulation.

1 Introduction

Many E-commerce sites offer hundreds of thousands or even millions of products for sale. Although various products can be obtained on the web, a customer has to invest time and effort to find satisfactory products. A recommender system provides personalized recommendation for each customer to reduce the problem [1, 2]. One of the most successful recommendation technologies is Collaborative Filtering [5, 6, 7], and is used in many websites[3]. A recommender system based on collaborative filtering try to predict a *utility* of products for a target user by using the ratings of products previously received from other users. The utility denotes satisfaction level to a product by the user.

In order to make accurate recommendations, the system has to first learn the preference or interests of the target user from the ratings given by the user. When

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1112–1121, 2007.

there are few ratings, the system cannot provide beneficial recommendation. This problem is known as *new user problem* or *cold start problem*[4]. Therefore, each user has to put in a certain number of ratings to the system, especially; this problem is serious in initial operation. One of the simplest solutions of the problem is to recommend the most popular product in all users as a substitute for personalized recommendation. However, it is difficult to estimate that how much ratings should be gathered, because, a recommendation effect is influenced by a characteristics of the community, i.e. the number of users and a bias of users' preference are important to a recommendation methodology.

The purpose of this paper is to evaluate the effect of community characteristics on recommender system, and we evaluate the effect using multi-agent simulation. In this paper, we focus on the number of users and a distribution of users' preference.

This paper is organized as follows: Section2 describes model settings based on multi-agent simulation. Then, Section3 explains procedure of experiments using the simulation model, and Section 4 shows the results and discussion. Finally, Section 5 concludes the paper.

2 Simulation Model

In order to understand the recommendation process, a multiagent-based simulation model was built and applied. In this model, it was assumed that users who have different interest, request recommendation for certain products from the system. Each user has his/her own preference, and each product has its own feature. Generally, if a certain user finds the product suitable for his/her preference, the user will reach high satisfaction level. The degree of such satisfaction can be called *utility*. Thus, the role of a recommender system can be rephrased as recommending certain products for the target user so that the utility will be improved.

Generally, a recommender system receives information from a user about the products the user is interested in. If the system obtains utility directly, it can accurately measure a satisfaction level. However, the only collectible information by the system is *rating* in this model. The rating is coming from the utility, i.e. the utility, which exists in the user's mind, is converted into rating in order to be able to put in the recommender system as feedback information. If the correspondence between utility and rating is optimal, the system will give the most accurate information.

There are many websites collecting ratings about products, using a form interface such as two-choice question (like or dislike) or five-grade rating (from one star to five stars). In this paper, five-grade rating was adopted as the form interface. Details of the model are shown below.

2.1 Model Components

The simulation model has three components: user, product and recommender system. This model can be formulated as follows: Let $U = \{u_i | i = 1, 2, ..., N_u\}$

be the set of all users, let $C = \{c_j | j = 1, 2, ..., N_c\}$ be the set of all products and let RS be a recommender system. The preference of each user u_i is represented by the vector $\mathbf{p_i} = (p_{i1}, p_{i2}, \ldots, p_{iN_f})$, and the feature of each product c_j is represented by the vector $\mathbf{v_j} = (v_{j1}, v_{j2}, \ldots, v_{jN_f})$, where N_f is the dimension of $\mathbf{p_i}$ and $\mathbf{v_j}$.

Intuitively, feature values (elements of the $\mathbf{v_j}$) may be related to shape of product, color of product, etc. Each element of vector $\mathbf{p_i}$ denotes the most ideal product for user u_i. When, it comes to define utility, the distance between the preference vector $\mathbf{p_i}$ and the feature vector $\mathbf{v_j}$ may be calculated. If the distance is small, the utility is high. Thus, the dimension of the two vectors should be equal.

2.2 Utility

Utility function $f_{utility}$ is a function to calculate the utility value s_{ij} from the preference vector $\mathbf{p_i}$ and the feature vector $\mathbf{v_j}$. The function is formulated as follows:

$$s_{ij} = f_{utility}(\mathbf{p_i}, \mathbf{v_j})$$

This function can be designed in several ways; however, there are certain rules that have to be followed: the function has to be monotonically decreasing function, and it has to be inversely proportional to the distance of the two vectors. The most important factor for the user u_i is to find a product with the closest utility to his/her preference. This information can be obtained from calculating the differences between utilities in the area near the preference vector. In this paper, an exponential function is used to accentuate of the utilities, and that can be formulated as follows:

$$\begin{aligned} f_{utility}(\mathbf{p_i}, \mathbf{v_j}) &= exp\big(-\alpha||\mathbf{p_i} - \mathbf{v_j}||\big) \\ &= exp\Big(-\alpha\sqrt{\sum_{k=1}^{N_f}(p_{ik} - v_{jk})^2}\Big) \end{aligned}$$

where α is adjusted by the value range of the vectors. α has to be a positive constant, and in this experiment, it was set as $\alpha = 0.5$. The value range of the function is $(0, 1]$.

2.3 Rating

In this model, let R be the set of all product ratings by users, and let rating $r_{ij} \in R$ be the rating for product $c_j \in C$ by user $u_i \in U$. Even if different users have same utilities about certain product, different ratings can be generated. However, it is preferable to decrease this difference as much as possible, for instance by presenting detailed explanation on a website about the rating process. In this model, an ideal setting is assumed, i.e. every user decides his/her rating based on the same rating process f_{rating}.

As described above, five-grade rating was used in this model. Thus, the function f_{rating} has four thresholds. Users should rate uniformly, without biasing to

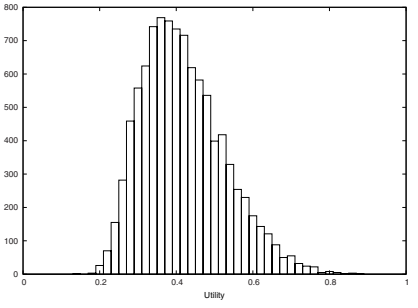


Fig. 1. Utility distribution when preference vector $\mathbf{p_i}$ and feature vector $\mathbf{v_j}$ are given at a uniform random number

a low or high rating. This histogram on figure1 shows the distribution of utility s_{ij}, when preference vector $\mathbf{p_i}$ and feature vector $\mathbf{v_j}$ are given at a uniform random number.

The utilities on this histogram were divided into five equal areas, and the boundary values were utilized as the thresholds of function f_{rating}. Accordingly, function f_{rating} is defined in this model as follows:

$$r_{ij} = f_{rating}(s_{ij}) = \begin{cases} 1 & (s_{ij} \leq 0.33) \\ 2 & (0.33 < s_{ij} \leq 0.38) \\ 3 & (0.38 < s_{ij} \leq 0.44) \\ 4 & (0.44 < s_{ij} \leq 0.51) \\ 5 & (0.51 < s_{ij}) \end{cases}$$

2.4 Recommendation Algorithm

The recommendation problem is reduced to the problem of estimating ratings for the products that have not been seen by a user [4]. In most cases, users rate only a little subset of items, in other words, most of the rating r_{ij} is unapparent. Therefore, the role of RS is to estimate unapparent ratings. The product with the highest estimation value of rating will be offered by recommender systems. In this paper, the following recommendation algorithms were experimented.

Random Recommendation: Random Recommendation algorithm provides the random estimations of the unapparent ratings. In fact, the system recommends a randomly-selected product to the target user. This algorithm is the basis of evaluation of other recommendation algorithms.

Popular Products Recommendation: Popular Products Recommendation recommends the product that receives the highest popularity from all users. The system calculates the mean value of the evaluation of all products, and the product with the highest mean value will be recommended to the target user.

If two or more products have same mean value of the ratings, one of them is chosen at random, and recommended. Mostly, a lot of users are recommended the same product, as the number of ratings increase.

Collaborative Filtering: Collaborative Filtering (CF) estimates the unapparent ratings based on the ratings of other users [3, 6]. When the system estimates ratings for the target user, firstly, CF organizes a set of users who have a history of agreeing with the target user using statistical techniques. In this operation, CF calculates similarities between the target user and the others.

Generally, the similarity between two users is defined by the correlation coefficient of the item rating which either of the users evaluated as follows:

$$sim(u_i, u_j) = \frac{\sum_{a \in C_i \cap C_j} (r_{ia} - \overline{r}_i)(r_{ja} - \overline{r}_j)}{\sqrt{\sum_{a \in C_i \cap C_j} (r_{ia} - \overline{r}_i)^2} \sqrt{\sum_{a \in C_i \cap C_j} (r_{ja} - \overline{r}_j)^2}}$$

$$\text{where} \quad \overline{r}_i = \frac{1}{|C_i|} \sum_{a \in C_i} r_{ia}$$

C_i is the set of products in which user u_i was already rated, and C_{ij} is also the set of products in which both user u_i and user u_j were already rated. If C_{ij} is an empty set, the similarity will be calculated as $sim(u_i, uj) = 0$.

After every similarity between two users has been calculated, an unapparent ratings $\widehat{r}_{i\alpha}$ for a product c_α by user u_i can be described as follows:

$$\widehat{r}_{i\alpha} = \widetilde{r}_i + \frac{\sum_{j \in U_\alpha} sim(u_i, u_j)(r_{j\alpha} - \overline{r}_j)}{\sum_{j \in U_\alpha} |sim(u_i, u_j)|}$$

where U_α is the set of users who rated the product c_α, and $\widetilde{r}_i$ is the average of all ratings that user u_i ever rated as follows:

$$\widetilde{r}_i = \frac{1}{|C_i|} \sum_{a \in C_i} r_{ia}$$

As a result, the system recommends the product c_α with the maximum of rating estimate $\widehat{r}_{i\alpha}$ to user u_i.

In this model, RS recommends a certain product using the algorithm for unapparent ratings $\widehat{r}_{i\alpha}$ based on the correlation coefficient. However, in case of too less ratings, all similarities of user u_i cannot be calculated. Therefore, the rating estimate is defined as $\widehat{r}_{i\alpha} = 3$, and RS makes recommendation at random from among the products which have same estimate.

3 Experiments

3.1 Simulation Procedure

Recommender systems based on CF can improve the accuracy of the recommendation as the number of ratings increases. Even if the system can utilize only few ratings, an effective recommendation will be provided as much as possible. When evaluating RS, the relationship between the number of ratings and the accuracy (utilities of all users) should be revealed.

In the following experiments, N_u is a number of users and N_c is a number of products. *RS* recommends a product each time a rating was received from a user, and then the user rates the recommended product for *RS*. This interaction is repeated until all of N_u users rate all of N_c products. This operation can be described more in detail as follows:

Step 1: Each user picks up one product among all products at random. Then, a utility of the product is calculated, and a rating obtained from the utility is put into RS by each user.

Step 2: *RS* estimates the ratings of the products, which have not been rated yet, using the rating already put in by the users. As described above, three types of recommendation algorithm are utilized for this estimation.

Step 3: The product which has the highest rating estimation is recommended to a target user.

Step 4: Each user calculates a utility of the product recommended by *RS*. Since an average of the utilities can be perceived as recommendation accuracy, the average is named *UtilityAverage*. After the rating obtained from the utility is put into RS, the system returns to step2.

3.2 Experimental Settings

Distribution of the Preference Vector: The preference vector $\mathbf{p_i}$ represents the most ideal product of the user. Therefore, the distribution of the vector $\mathbf{p_i}$ intuitively denotes the trend or the vogue among the users. Each element of the vector $\mathbf{p_i}$ is set as random numbers according to the following distribution. Four types of the distribution were applied to investigate an effect of the recommendation by the difference of the distribution. Note that each element of the vector is mutually independent.

- Uniform distribution of range [-1, 1]
- Normal distribution $N_1(\mu = 0, \sigma^2 = 0.5)$
- Normal distribution $N_2(\mu = 0, \sigma^2 = 0.2)$
- Two-peaks distribution (A distribution obtained by randomly selecting either one of the following two normal distributions: $N(\mu = -0.5, \sigma^2 = 0.2)$ or $N(\mu = 0.5, \sigma = 0.2)$)

Uniform distribution represents a situation without trend of the preference, and normal distribution represents a situation where the trend exists. Particularly,

the trend is strong when the variance is small in the normal distribution. Furthermore, Tow-peaks distribution represents a situation that the users form several group by their preference. The number of groups is calculated by 2^d where d is the dimension of the vector $\mathbf{p_i}$, because random numbers are independently given to each element.

Constants: Table 1 shows the experiment constants that we used.

Table 1. Experiment Constants

Number of users N_u	100, 500, 1000, 5000, 10000
Number of products N_c	1000
Dimension of vector p_i	5
Dimension of vector v_j	5
α of utility function $f_{utility}$	0.5

4 Results and Discussion

Distribution of the Feature Vector: In this simulation, various products are provided as the recommendation candidates, for investigating an effects of user preference on that recommendation performance. If there is bias in the distribution of feature vectors $\mathbf{v_j}$, it also effects the recommendation result. However, in this experiments, we focus on the effect of the distribution of preference vector. Thus, we eliminate the effect of the distribution of $\mathbf{v_j}$ on the recommendation result by providing the feature vector $\mathbf{v_j}$ at random. More formally, the vector $\mathbf{v_j}$ is set as numbers according to the uniform distribution of range$[-1, 1]$.

4.1 Experiments with Each Recommendation Algorithm

The following figures (from figure2 to figure5) show experimental results by each recommendation algorithm, in four types of distribution of the preference vector $\mathbf{p_i}$. The figures are results in case of $N_u = 1000$, and these figures were made by one-shot experiment. However, if the number of users increases, fluctuation at each experiment is few, because the graphs show UtilityAverage. An advantage of the popular products recommendation and the collaborative filtering can be observed by comparing with the random recommendation.

CF is equal to the random recommendation when there are few ratings, and it begin to produce the effectiveness from the vicinity of 30 ratings. The effect of CF is declined in the latter half of the figures, because it previously recommend certain products that matches to the users preference. Popular Product Recommendation produces better UtilityAverage than CF in early step, and a cross point between the line of Popular Product Recommendation and CF is about 30 ratings.

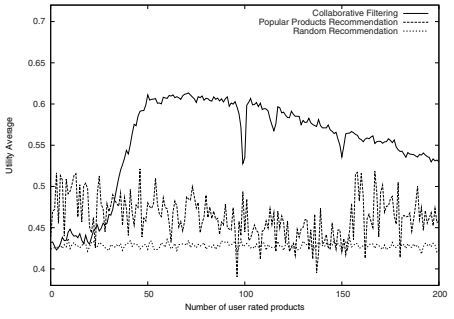


Fig. 2. Uniform distribution

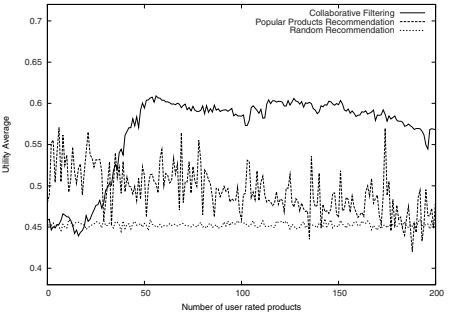


Fig. 3. Normal distribution $N_1(\sigma^2 = 0.5)$

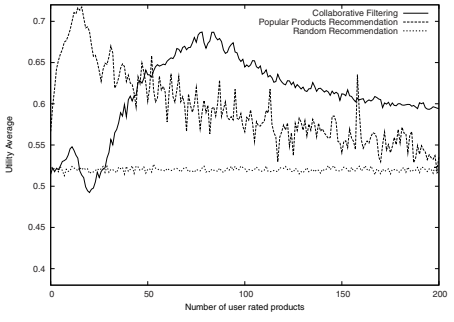


Fig. 4. Normal distribution $N_2(\sigma^2 = 0.2)$

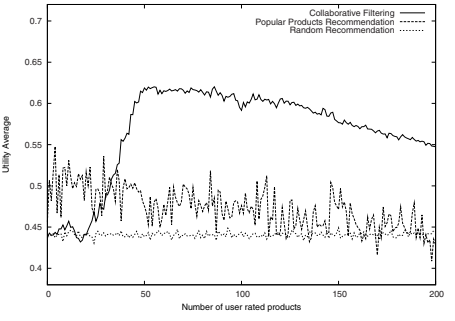


Fig. 5. Two-peaks distribution

When we compare the difference by the distribution of the preference vector, first of all, UtilityAverage of random recommendation is different in four figures. The figures show that, if the distribution of the preference vector is N_2, UtilityAverage of random recommendation is most high, because UtilityAverage rises when the vector is gathered in the center. Moreover, in the N_2 distribution, Popular Products Recommendation produce high UtilityAverage, because entire popularity was gained by several products, which have a feature vector toward near the center. However, CF is the highest result in the latter half of any figures.

4.2 Relationship Between Number of Users and Recommendation Accuracy

The following figures (from figure6 to figure9) shows the experimental results of CF based recommendation in various numbers of users. Each figure illustrates the result in the situation of each distribution of preference vector $\mathbf{p_i}$. The figures were also made by one-shot experiment. It is necessary to repeat similar experiments to obtain a more accurate experimental result. However, the tendency to the recommendation accuracy in an increase in the number of users can be obtained from these results.

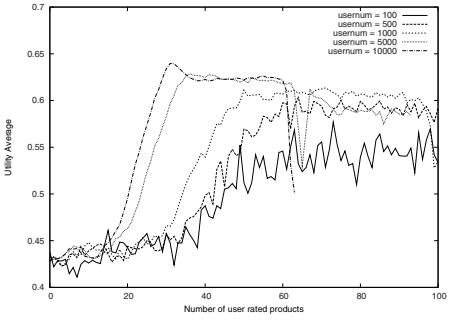


Fig. 6. Uniform distribution

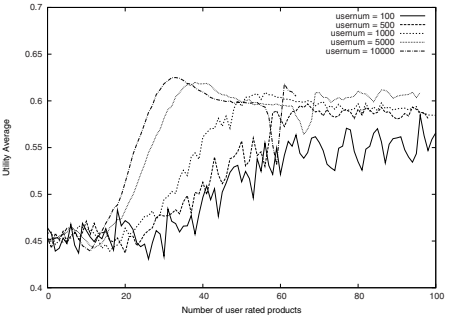


Fig. 7. Normal distribution $N_1(\sigma^2 = 0.5)$

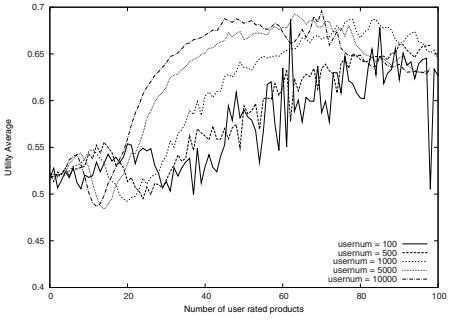


Fig. 8. Normal distribution $N_2(\sigma^2 = 0.2)$

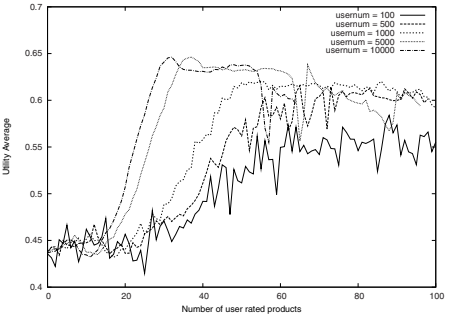


Fig. 9. Two-peaks distribution

As discussed previously, CF in early step is equal to a random recommendation. We should compare the recommendation effectiveness by difference of UtilityAverage between the value at 0 rating and the maximum value.

Each figure indicates that if there are a lot of users, UtilityAverage will be high; moreover, an effective recommendation in a less ratings is possible. For instance, if 10,000 users use the system and each user inputs 20 ratings, CF provides certain effective recommendation. However, even if the system has a lot of users, CF cannot recommend preferable products, when each user rates only less than about 20 products. In this case, the system should provide popular product recommendation together.

In addition, when the number of users is little, the UtilityAverage is not high, even if many ratings are given by each user. In a word, it is important in the improvement of recommendation accuracy to gather many users.

5 Conclusion

In this paper, we built a multi-agent based simulation model for investigation of the effectiveness of a recommendation system. Especially, we investigated collaborative filtering which is used in many websites today.

Recommender system recommends certain products based on a lot of product's ratings which were previously rated by the users. Thus, the accuracy of CF based recommendation depends on a community characteristics, such as the number of users or a distribution of users' preference. In this research, we examined the effect of community characteristics on the recommender system. And we investigated the difference between popular product recommendation and CF.

The results show that, popular product recommendation provides higher utility than CF, if there are few ratings. And the CF overtakes if a lot of ratings are utilized. Moreover, it was confirmed that the recommendation accuracy is improved when there are a lot of users.

References

1. Resnick, P., Varian, H.: Recommender Systems. Comm. of the ACM 40(3), 56–89 (1997)
2. Reategui, E., Campbell, J.A., Torres, R.: Using Item Descriptors in Recommender Systems. In: AAAI Workshop on Semantic Web Personalization, San Jose, USA (2004)
3. Linden, G., Smith, B., York, J.: Amazon.com Recommendations: Item-to-Item Collaborative Filtering. IEEE Internet Computing (January / February 2003)
4. Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)
5. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of Recommendation Algorithms for E-Commerce. In: Proceedings of the 2nd ACM conference on Electronic commerce, Minneapolis, Minnesota, United States, October 17-20, pp. 158–167 (2000)
6. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In: CSCW 1994. Conference on Computer Supported Cooperative Work (Chapel Hill, 1994), pp. 175–186. ACM, New York (1994)
7. Good, N., Konstan, J.B., Borchers, J.A., Sarwar, B., Herlocker, J., Riedl, J.: Combining Collaborative Filtering with Personal Agents for Better Recommendations. In: AAAI 1999. Proceedings of the 1999 Conference of the American Association for Artificial Intelligence, pp. 439–446 (1999)
8. Tung, H.-W., Soo, V.-W.: A Personalized Restaurant Recommender Agent for Mobile E-Service. In: EEE 2004. Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (2004)

Exploring Quantitative Evaluation Criteria for Service and Potentials of New Service in Transportation: Analyzing Transport Networks of Railway, Subway, and Waterbus*

Keiki Takadama[1], Takahiro Majima[2], Daisuke Watanabe[2], and Mitsujiro Katsuhara[3]

[1] The University of Electro-Communications, 1-5-1, Chofugaoka, Chofu, Tokyo 182-8585 Japan
keiki@hc.uec.ac.jp
[2] National Maritime Research Institute, 6-38-1, Shinkawa, Mitaka, Tokyo 181-0004 Japan
{majy,watanabe}@nmri.go.jp
[3] Hokkaido Intellect Tank, 1-4-10, Nishishimbashi, Minatoku, Tokyo, Japan
kat-151@mail.bbexcite.jp

Abstract. This paper explores *quantitative evaluation criteria* for service and *potentials* of new service from the transportation viewpoint. For this purpose, we analyze transport networks of railway, subway, and waterbus, and have revealed the following implications: (1) *efficiency* criterion proposed by Latora [7,8] and *centrality* criterion in the complex network literature can be applied as quantitative evaluation criteria for service in a transportation domain; and (2) new services are highly embedded among networks, *i.e.*, the analyses of the combined networks have the great potential for finding new services that cannot be found by analyzing a single network.

Keywords: service science, evaluation criteria, transportation, complex network.

1 Introduction

Service science, management and Engineering (SSME) [4,5] is a new research field that stresses an importance of addressing *service* from the viewpoint of *science* in order to clarify effects of services based on experience and intuition. This approach has the great potential of utilizing or bringing out the hidden experience and vague intuition in human. In spite of this potential, however, the

* Paper submitted to the 8th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL'07).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1122–1130, 2007.

following significant problems have not yet solved in SSME: (1) it is generally difficult to evaluate effects of service precisely, *i.e.*, services is hard to be measured quantitatively; (2) it is also difficult to find new services which are useful and have a high value in a market.

To tackle these problems, this paper aims at exploring quantitative evaluation criteria for service and proposing an approach to find new services. As the first step towards our goal, this paper starts by narrowing arguments down to the transportation domain such as railway, subway, and waterbus, and analyze characteristics of the network from the viewpoint of a *complex network* [1,11,12]. We focus on transport networks because transportation is indispensable service in our life.

This paper is organized as follows. The next section starts by explaining major measures in a complex network literature, and Section 3 describes the outline of the transport network analysis. The quantitative evaluation criteria for service and an approach to find new services are discussed in Section 4. Finally, our conclusions are given in Section 5.

2 Measures

Measures characterizing networks are divided into two types, *weighted* or *unweighted* networks. According to Sienkiewicz, the former network is the physical infrastructure network in Euclidean space, while the latter network is the unipartite network [10]. In these two network types, many measures were proposed to evaluate networks. Some of measures are described below.

2.1 Measure in Weighted Network: Efficiency

The measure called *efficiency* [7,8] was proposed to consider the physical distance for weighted networks, which is defined as the following equations.

$$E(\mathbf{G}) = \sum_{i \neq j \in \mathbf{G}} \varepsilon_{ij} \tag{1}$$

$$\varepsilon_{ij} = \frac{1}{d_{ij}} \tag{2}$$

In these equations, $\mathbf{G}$ is a network, ε_{ij} is the efficiency that is inversely proportional to the d_{ij} representing the geographical distance along the shortest path between vertex i and j (if there is no path to connect vertex i and j, $d_{ij}$is infinity). Global efficiency, E_{glob}, and local efficiency, E_{loc}, are obtained from Eqs. (3)(4) respectively with above E.

$$E_{glob} = \frac{E(\Gamma)}{E(\Gamma_{id})} \tag{3}$$

$$E_{loc} = \frac{1}{N} \sum_{i \in \Gamma} \frac{E(\Gamma(\nu_i))}{E(\Gamma_{id}(\nu_i))} \tag{4}$$

In these equations, Γ and $\Gamma(\nu_i)$ denotes the whole network with N vertices and a sub-graph composed of neighbor vertices of vertex i, respectively. Subscript id means idealized network comprised of complete graph among all vertices belonging. Thus, d_{ij} of any ideal network becomes direct distance in Euclidean space. It is clear by definition that global or local efficiency ranges from 0 to 1. The efficiency becomes 1 when all pairs of vertices are connected by the straight links in the Euclidean space.

2.2 Measure in Un-weighted Network: Centrality

Among several measures of *centrality* [2,3,6], *eigenvector centrality* is a measure of an importance of a vertex in a network in terms of topology. This definition becomes eigenvector equation as is shown in Eq.(5) and the vector $\mathbf{x}$ is called as eigenvector centrality. The λ and $\mathbf{x}$ in Eq.(5) stand for the maximum eigenvalue and corresponding eigenvector of an adjacent matrix, $\mathbf{A}$, respectively. Applying the eigenvector centrality to the adjacent matrix, the rank of centrality of stations can be obtained. Since the un-weighted network is comprised of complete graph based on the lines, the rank evaluates an importance of stations from the viewpoint of topology of train or bus lines.

$$\mathbf{x} = \frac{1}{\lambda}\mathbf{A}\mathbf{x} \tag{5}$$

3 Analysis

3.1 Overview

Our previous research [9] analyzed five transport networks, *i.e.*, one railway, three subways, and one hypothetical waterbus line in terms of complex network using the measures described in the previous section. This analysis was typically performed around Tokyo Metropolitan, because Tokyo is the most populated area in Japan. For subway, in particular, we analyzed not only the subway network in Tokyo but also those in Osaka and Nagoya, both of them are major cities in Japan.

Furthermore, the location of stations and links in the railway and subways were based on the real date, while those of waterbus is hypothetical due to the fact that its network does not exist now. We, however, consider the waterbus because it provides an efficient transport network from the viewpoint of modal shift or reducing the commuter rushes. To show such effects, several projects are under way [13], which employs the combination of the river system and ships as a substitute of road network suffered from chronic traffic congestion.

3.2 Evaluation Criteria

As evaluation criteria, we employed the measures described in Section 2, *i.e.*, (1) global and local efficiencies from the weighted network viewpoint; and (2)

eigenvector centrality from the un-weighted network viewpoint. The detailed meaning of these criteria are summarized as follows. Note that geographical distance between two stations is employed as the weight in the network for the efficiently criterion[1].

- **Global efficiency** measures the path length of any pair of stations in the network by comparing that in the ideal network in which any pair of stations is connected by straight line in the Euclidean space. This measure evaluates the network from the viewpoint of the *shortest path length*.
- **Local efficiency** measures the number of alternative paths of any pair of stations in the network by comparing that in the ideal network. This measure evaluates the network from the viewpoint of *redundancy of paths*.
- **Eigenvector centrality** measures the numbers of connected stations, which includes not only directly connected stations but also near stations (*e.g,*, stations which are closed to connected stations).

Finally, efficiency is calculated by the Warshall-Floyd shortest path algorithm applying adjacent matrix, while the eigenvector centrality is calculated by Jacobi method.

4 Discussion

4.1 Efficiency

Unit network analysis. Our previous research [9] calculated the efficiency as shown in Table 1. In this table, N, E_{glob}, E_{loc}, RW, SW, and WB indicate the number of stations, the global efficiency, the local efficiency, railway, subway, and waterbus, respectively. In addition to five transport networks and one combined network of JR and subway in Tokyo, results for the subway in Boston obtained from literatures [7,8] is also described in the table. Combination network above means the transport networks of JR and subway in Tokyo, those of which are combined at the stations. There are 37 stations that JR railway and subway in Tokyo have in common.

From Table 1, the global efficiency, E_{glob}, indicates the large efficiency around 0.7, which means that the public transport networks in real world is about 30% less efficient than the ideal network. The local efficiencies, E_{loc}, on the other hand, are extremely small except for waterbus network. Specifically, the local efficiencies of subway in Osaka and Nagoya are 0, which means that triangle comprised of three adjacent stations does not exist at all, *i.e.*, the networks of subway in Osaka and Nagoya does not have redundant paths. When such triangle exists, passengers can circumvent the station or the link with another path in triangle even if one of three stations or links composing a triangle is

[1] The trip time can be considered another important factor as weight. However, the transferring time from one line to another line is not the same according to the type of train, such as express train or local train. From this fact, our previous research employed the geographical distance not trip time.

Table 1. Global and local efficiencies in five transport networks

Mode*)	N	E_{glob}	E_{loc}
JR(RW)	371	0.78	0.033
Tokyo(SW)	211	0.70	0.024
Osaka(SW)	100	0.72	0
Nagoya(SW)	82	0.80	0
Boston(SW)[7][11]	124	0.63	0.030
JR(RW)+Tokyo(SW)	544	0.75	0.030
Tokyo(WB)	24	0.77	0.20

*) RW:Railway, SW:Subway, WB:Waterbus

damaged. Thus, the triangle in the transport network leads to a redundancy. In comparison with these small E_{loc}, that in waterbus network is relatively large, which indicates that waterbus network has a high redundancy.

It should be noted here that a high redundancy has the great potentials of providing passengers *many services*, *e.g.*, passengers can select alternative paths in terms of minimizing time or costs. Alternative paths also contributes to providing business chances in transfer areas, *e.g.*, shops or stores in transfer areas. However, these services are based on high global efficiency because the network with the low global efficiency does not have many direct routes which prevent passengers from selecting such network. From this viewpoint, the following implications are revealed: (1) both high global and local efficiencies are required to increase business chances; and (2) the waterbus network that has high global and local efficiencies has the potentials of providing business chances in comparison with other public transport networks.

Combined network analysis. Previous section shows the high potential of the waterbus network, but what should be noticed here is that the business chances provided by this potential depends on the number of passengers. Considering the fact that the numbers of passengers in railway and subway is quite larger than that in waterbus, the waterbus network is hard to provide *big* business chances. From this fact, this section considers the network that combines the railway and/or subway networks with the waterbus network.

Our previous research [9] assumed that the waterbus stations can be identical with railway or subway stations when the distance between them is less than $500m$. There were identical waterbus stations with stations of three in JR, six in subway in Tokyo, and seven in JR and subway combined network. Calculated measures on all combinations with the waterbus network are summarized in Table 2. This table indicates that the global efficiency, E_{glob}, is not affected by the combination with the waterbus network, while the local efficiency, E_{loc}, in the subway network and the combined network of JR and subway increases, even though the number of waterbus stations is very small. For example, E_{loc} in the subway network is 0.024 and it becomes 0.031 when the waterbus network connects.

Table 2. Global and local efficiencies with or without combining waterbus network

Mode	N	Without WB		With WB	
		E_{glob}	E_{loc}	E_{glob}	E_{loc}
JR	371	0.78	0.033	0.78	0.033
SW	211	0.70	0.024	0.70	0.031
JR+SW	544	0.75	0.030	0.75	0.033

This results revealed that an combination of the waterbus network with other transport ones has the potential of increasing business chances because the waterbus network contributes to increasing E_{loc} with keeping a high E_{glob}.

4.2 Centrality

Network analysis without waterbus. Table 3 shows the top 10 stations of a *centrality* in the combined network of JR and subway in Tokyo. In this table, k and x represent *degree* (*i.e.*, the number of links connected to a station) and *centrality* of each station, respectively. Although the positive correlation is observed between degree and centrality, it does not need that the station with the higher degree has the higher rank in terms of centrality. Since the stations, such as Higashi-Nakano or Ryogoku, are not major stations (according to the census data, the ranks of these stations in the number of users are lower than 100), people living in and around Tokyo may feel peculiar about the rank in Table 3. The reason is that stations in the high rank have lines connecting to a number of another lines like Sobu and Oedo lines where the numbers of transfer stations of the two lines are the second and third highest values among the lines. Thus, passengers at stations with a high centrality can reach to most stations without changing trains.

From this analysis, a high centrality has the great potentials of increasing business chances because passengers do not want to increase transfer times but

Table 3. Eigenvector centrality in the top 10 stations

Rank	Station Name	x	k
1	Shinjuku	0.200	164
2	Tokyo	0.171	167
3	Akihabara	0.161	120
4	Yoyogi	0.158	97
5	Iidabashi	0.148	126
6	Shimbashi	0.127	116
7	Ueno	0.120	126
8	Higashi-Nakano	0.115	71
9	Ryogoku	0.115	71
10	Kanda	0.115	85

want to select stations that can reach to most stations without changing trains. This suggests that all stations in Table 3 have the potentials of providing business chances. However, stations except for Higashi-Nakano or Ryogoku have been already full growth, which means that it is difficult to find other business chances in such stations. In comparison with these stations, Higashi-Nakano or Ryogoku stations are important for business chances. From this viewpoint, the following implications are revealed: (1) the high centrality is required to increase business chances; and (2) stations that has the high centrality but not major or full growth (such as Higashi-Nakano or Ryogoku) have the potentials of providing business chances.

Network analysis with waterbus. For the same reason discussed in Section 4.1, this section also consider the network that combines the railway and/or subway networks with the waterbus network. Table 4 summarizes the centrality of stations to which the waterbus connects. The centrality of some stations improves by several dozen percent and the rank of the stations becomes higher drastically. For example, the centrality of Honjoazumabashi station in the subway network increases by 45% and its rank improves from 175 to 86 owing to the waterbus network.

Table 4. Eigenvector centrality with or without combining waterbus network

	JR		JR+WB		x
Station Name	x	Rank	x	Rank	Increase
Etchujima	0.035	158	0.038	156	7%
Ryogoku	0.085	51	0.086	39	1%
Asakusa	0.011	227	0.013	218	25%

	SW		SW+WB		x
Station Name	x	Rank	x	Rank	Increase
Ryogoku	0.134	31	0.137	16	2%
Asakusa	0.041	79	0.051	54	22%
Hamacho	0.024	162	0.029	88	24%
Honjoazumabashi	0.020	175	0.030	86	45%
Morishita	0.149	11	0.151	8	1%
Higashi-ojima	0.024	166	0.029	87	25%

	JR+SW		JR+SW+WB		x
Station Name	x	Rank	x	Rank	Increase
Etchujima	0.025	203	0.031	191	25%
Ryogoku	0.115	9	0.118	8	3%
Asakusa	0.041	140	0.047	131	14%
Hamacho	0.016	353	0.021	253	28%
Honjoazumabashi	0.015	370	0.020	260	36%
Morishita	0.065	78	0.068	74	5%
Higashi-ojima	0.016	357	0.021	252	28%

This results revealed that an combination of the waterbus network with other transport ones has the potential of increasing business chances because the waterbus network contributes to increasing the centrality of stations to which the waterbus line connects.

4.3 Quantitative Evaluation Criteria and New Services

From the above analyses, the following implications have revealed:

- **Quantitative evaluation criteria:** The global and local efficiencies and centrality can be applied as quantitative evaluation criteria for service in a transportation domain. Specifically, the analyses suggests that both high global and local efficiencies and high centrality contribute to increasing business chances. This is because (1) the high global and local efficiencies provide not only mostly direct routes that promote passengers to select them but also alternative paths that can be selected by passengers in terms of minimizing time or costs; and (2) high centrality clarifies the stations that enable passengers to reach to most stations without changing trains. From the analyses, the waterbus network that has high global and local efficiencies has appeal for business from the efficiency viewpoint, while stations that has the high centrality but not major or full growth (such as Higashi-Nakano or Ryogoku) also have appeal for business from the centrality viewpoint.
- **An exploration of new services:** New services are highly embedded among networks, *i.e.*, the combined networks have the great potential for finding new services that cannot be found by analyzing a single network. Specifically, an combination of the waterbus network with other transport ones has the potential of increasing business chances because the waterbus network contributes to not only increasing E_{loc} with keeping a high E_{glob} but also increasing the centrality of stations to which the waterbus line connects. This is an approach to explore new services. Although an advertisement of the connection between waterbus and other transportation or a development of more smooth connection are indispensable to acquire business chances, new services can be introduced in the process of combining the waterbus network with other transport ones.

5 Conclusions

This paper explored *quantitative evaluation criteria* for service and *potentials* of new service from the transportation viewpoint. For this purpose, we analyzed transport networks of railway, subway, and waterbus, and have revealed the following implications: (1) *efficiency* criterion proposed by Latora [7,8] and *centrality* criterion in the complex network literature can be applied as quantitative evaluation criteria for service in a transportation domain. Specifically, both high global and local efficiencies and high centrality contribute to increasing business chances; and (2) new services are highly embedded among networks, *i.e.*,

the analyses of the combined networks have the great potential for finding new services that cannot be found in the single network. In our analyses, an combination of the waterbus network with other transport ones has the potential of increasing business chances.

However, these results have only been obtained from one domain, a transportation domain. Therefore, further careful qualifications and justifications, including the analysis of the results of other domains, are needed to generalize our results. Such important directions must be pursued in the near future in addition to the following future research: (1) an exploration of new quantitative evaluation criteria for service science not by employing measures that have been proposed so far; (2) an investigation of both efficiency and centrality introducing the flow in the network, *i.e.*, the number of passengers; (3) an investigation of effects of different hypothetical waterbus networks clarifying differences of features between railways and waterbus; and (4) an exploration of conditions of the networks that increase the global and local efficiencies and centrality.

References

1. Barabási, A.-L., Albert, R.: Emergence of Scaling in Random Networks. Science 286, 509–512 (1999)
2. Borgatti, S.P.: Centrality and Network Flow. Social Networks 27, 55–71 (2005)
3. Brin, S., Page, L.: The Anatomy of a Largescale Hypertextual Web Search Engine. Computer Networks and ISDN Systems 30(1–7), 107–117 (1998)
4. Chesbrough, H.: Towards a new science of services. Harvard Business Review 83(2), 43–44 (2005)
5. Chesbrough, H., Spohrer, J.: A research manifest for services science. Communications of the ACM 49(7), 35–40 (2006)
6. Freeman, L.C.: Centrality in Social Networks: Conceptual Clarification. Social Networks 1, 215–239 (1979)
7. Latora, V., Marchiori, M.: Efficient Behavior of Small-World Networks. Physical Review letters 87(19), 198701 (2001)
8. Latora, V., Marchiori, M.: Is the Boston Subway a Small-World Network? Physica A 314, 109–113 (2002)
9. Majima, T., Takadama, K., Watanabe, D., Katsuhara: Network Evolution Model for Route Design of Public Transport System and its Applications. International Transactions on Systems Science and Applications (ITSSA) 3(1) (to appear, 2007)
10. Sienkiewicz, J., Holyst, J.A.: Statistical Analysis of 22 Public Transport Networks in Poland. Physical Review E 72, 046127 (2005)
11. Watts, D.J., Strogatz, S.H.: Collective Dynamics of Small-World Networks. Nature 393, 440–442 (1998)
12. Watts, D.J.: Small Worlds. Princeton University Press, Princeton (1999)
13. www.ktr.mlit.go.jp/kyoku/river/pavilion/ship_08.htm (In Japanese)

Saw-Tooth Algorithm Guided by the Variance of Best Individual Distributions for Designing Evolutionary Neural Networks

Pedro Antonio Gutiérrez[1], César Hervás[1], and Manuel Lozano[2]

[1] Dept. of Computer Science and Numerical Analysis, University of Córdoba
Campus de Rabanales, building C2, 14004 - Córdoba, Spain
zamarck@yahoo.es, chervas@uco.es
[2] Dept. of Computer Science and Artificial Intelligence, University of Granada
E.T.S. Ingeniería Informática, 18071 - Granada, Spain
lozano@decsai.ugr.es

Abstract. This paper proposes a diversity generating mechanism for an evolutionary algorithm that determines the basic structure of Multilayer Perceptron (MLP) classifiers and simultaneously estimates the coefficients of the models. We apply a modified version of a recently proposed diversity enhancement mechanism [1], that uses a variable population size and periodic partial reinitializations of the population in the form of a saw-tooth function. Our improvement on this standard scheme consists of guiding saw-tooth reinitializations by considering the variance of the best individuals in the population, performing the population restart when the difference of variance between two consecutive generations is lower than a percentage of the previous variance. The empirical results over six benchmark datasets show that the proposed mechanism outperforms the standard saw-tooth algorithm. Moreover, results are very promising in terms of classification accuracy, yielding a state-of-the-art performance.

Keywords: Evolutionary algorithm, population reinitializations, saw-tooth algorithm, neural networks.

1 Introduction

Evolutionary Algorithms (EAs) are search algorithms based on the concept of natural selection. Among them, Evolutionary Programming (EP), originally proposed by L.J. Fogel [2], is a sthocastic optimization. One of the main characteristic of an EP algorithm is the absence of a direct codification, only working with their representation. Evolutionary artificial neural networks have been a key research area in the past decade, providing an interesting platform for optimizing both the weights and architecture of the network simultaneously. The problem of finding a suitable architecture and the corresponding weights of the network is a very complex task and this difficulty justifies the use of an evolutionary algorithm to design the structure and training of the weights (for a very interesting review on this subject the reader can consult [3]).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1131–1140, 2007.

Population size is one of the most important parameters affecting the robustness and computational efficiency of EAs. Small population sizes may result in premature convergence to nonoptimal solutions, whereas large population sizes require a considerable increase in computational effort. Although some authors try to estimate an optimal population size regarding the complexity of the problem [4], *variable population size* is a more interesting alternative [5], improving the EA capability of obtaining better solutions and reducing sensitivity in the election of different parameters. On the other hand, several methods using a constant census have been proposed in the literature that attempt to increase the diversity of the population and avoid premature convergence, including, among others, alternative mutation strategies [6], the well known Eshelman's CHC algorithm [7] or the micro Genetic Algorithm or μ-GA, suggested by Goldberg [8]. Both CHC and μ-GA algorithms are based on *periodical reinitializations* of population when the diversity drops below a threshold.

One methodology combining the effects of variable population size with periodical reinitializations is that proposed by Koumousis and Katsaras [1], which follows a saw-tooth scheme with a specific amplitude and period of variation. In each period, the population size decreases linearly and, at the beginning of the next period, randomly generated individuals are appended to the population. One of the major drawbacks of this approach is that both the amplitude and period of saw-teeth must be specified a priori.

In this paper, we propose an enhanced version of this saw-tooth EA, in which we guide the period of each saw-tooth according to the variance of the best individual fitnesses of the population. The underlying idea is that reinitializations should be performed when the best individuals have converged and no further optimization is being achieved, so randomly generated individuals can help the algorithm to explore new regions in the search space. We apply this proposed scheme over a previously implemented EP Algorithm [9], that evolves the weights and the structure of MLP Neural Network classifiers. In order to test the performance and suitability of our methodology, a comparative study using the original constant population EP algorithm, the standard saw-tooth EP algorithm and the variance guided saw-tooth EP algorithm has been applied to six datasets taken from the UCI repository [10]. The rest of the paper is organized as follows: Section 2 is dedicated to a description of the selected EP algorithm; Section 3 describes the implementation of the standard saw-tooth scheme and its application to the EP algorithm; the aim of Section 4 is to describe the proposed variance guided version of the saw-tooth scheme; Section 5 explains the experiments carried out; and finally, Section 6 shows the conclusions of our work.

2 Evolutionary Programming Algorithm

In this section, we present the EP algorithm used to estimate the parameters and the structure of the MLP neural networks models. The main objective of the algorithm is to design a neural network with optimal structure and weights for each classification problem tackled. The search begins with an initial population

of MLPs, to which a population-update algorithm is applied in each iteration. The algorithm shares many characteristics and properties with other previous algorithms like [11] and [12]. Individuals are subjected to the operations of replication and mutation. Crossover is not used due to its potential disadvantages in evolving artificial neural networks. With these features the algorithm falls into the class of EP.

We consider standard feed forward MLP neural networks with one hidden layer and we interpret the outputs of the neurons on the output layer from a probability point of view which considers the softmax activation function given by the following expression:

$$g_l(\mathbf{x}, \boldsymbol{\theta}_l) = \frac{\exp f_l(\mathbf{x}, \boldsymbol{\theta}_l)}{\sum_{j=1}^{J} \exp f_j(\mathbf{x}, \boldsymbol{\theta}_j)} \tag{1}$$

where J is the number of classes in the problem, $f_l(\mathbf{x}, \boldsymbol{\theta}_l)$ the output of the neuron j for pattern $\mathbf{x}$ and $g_l(\mathbf{x}, \boldsymbol{\theta}_l)$ the probability that pattern $\mathbf{x}$ has of belonging to class j. Considering this expression, the classification rule $C(\mathbf{x})$ of the MLP is the following:

$$C(\mathbf{x}) = \widehat{l}, \quad \text{where } \widehat{l} = \arg\max_l g_l(\mathbf{x}, \boldsymbol{\theta}_l), \text{ for } l = 1, 2, ...J \tag{2}$$

We define the Correctly Classified Rate by $CCR = (1/N) \sum_{n=1}^{N} (I(C(\mathbf{x}_n) = \mathbf{y}_n)$, where $I(\cdot)$ is the zero-one loss function. A good classifier tries to achieve the highest possible CCR in a given problem. The function used in the EP algorithm for obtaining the fitness of individuals is a strictly decreasing transformation of the cross entropy error function and is given by the following expression:

$$A(g) = \frac{1}{1 + l(\boldsymbol{\theta})} \tag{3}$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_J)$ and $l(\boldsymbol{\theta})$ is the cross entropy error function of the model and is obtained as:

$$l(\boldsymbol{\theta}) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{J} y_n^{(l)} \log g_l(\mathbf{x}_n, \boldsymbol{\theta}_l) = \tag{4}$$

$$= \frac{1}{N} \sum_{n=1}^{N} \left[-\sum_{j=1}^{J} y_n^{(l)} f_l(\mathbf{x}_n, \boldsymbol{\theta}_l) + \log \sum_{j=1}^{J} \exp f_l(\mathbf{x}_n, \boldsymbol{\theta}_l) \right]$$

The general framework of the EP algorithm is the following:

1. Generate a random population of size N_{P}.
2. Repeat until a maximum number of generations G is reached:
   (a) Apply parametric mutation to the best 10% of individuals. Apply structural mutation to the remaining 90% of individuals.
   (b) Calculate the fitness of every individual in the population.

(c) Add best fitness individual and best CCR individual of the last generation (double elitist algorithm).
(d) Rank the individuals with respect to their fitness.
(e) Best 10% of population individuals are replicated and substitute the worst 10% of individuals.

3. Select the best CCR individual and the best fitness individual in the final population and consider both as possible solutions.

As mentioned previously, fitness is a decreasing transformation of cross-entropy error. In general, the relationship between CCR and cross-entropy error strongly depends on the data base structure. Hence, regarding experimental results, using cross-entropy elitism is more suitable for some databases to result in a higher generalization accuracy, but using CCR elitism can be more appropriate for some other databases. For this reason, the EP algorithm returns both the best CCR and the best fitness individuals as solutions, the best approach for each problem being difficult to ascertain a priori. Parametric mutation is accomplished for each coefficient of the model with Gaussian noise, applying a standard simulated annealing process for accepting or rejecting modifications. On the other hand, we use five different structural mutations, similar to the mutations in the GNARL model [11]. The severity of mutations depends on the temperature $T(g)$ of the neural network model, defined by $T(g) = 1 - A(g)$ $0 \leq T(g) \leq 1$.

In order to define the topology of the neural networks, we consider three parameters: m, M_{E} and M_{I} . They correspond, respectively, to the minimum and maximum number of hidden nodes in the whole evolutionary process and the maximum number of hidden nodes in the initialization process. In order to obtain an initial population formed by models simpler than the most complex model possible, parameters must fulfil the condition $m \leq M_{\mathrm{E}} \leq M_{\mathrm{I}}$. More details about the EP algorithm can be consulted in [9], [13] and [14]. The EP algorithm was implemented using the Evolutionary Computation framework JCLEC [15] (`http://jclec.sourceforge.net`) and is available in the non-commercial java tool named KEEL (`http://www.keel.es`).

3 Standard Saw-Tooth Evolutionary Programming Algorithm

The scheme proposed in [1] is briefly summarized in this section. The Standard Saw-tooth algorithm utilizes a variable population size following a periodic scheme where a mean population size $\overline{n}$, an amplitude D and a period of variation T define the saw-tooth shape. Thus, at a specific generation t, the population size $n(t)$ is determined as:

$$n(t) = \mathrm{int}\left\{\overline{n} + D - \frac{2D}{T-1}\left[t - T \cdot \mathrm{int}\left(\frac{t-1}{T}\right) - 1\right]\right\} \tag{5}$$

where $int(\cdot)$ is the floor function. Therefore, $n(1) = \overline{n} + D$, $n(T) = \overline{n} - D$, $n(T+1) = \overline{n} + D$, etc.The selection of the $\overline{n}$, T and D parameters affects the

performance of the algorithm. In the paper cited, the optimum values of the T and D parameters of the saw-tooth are obtained experimentally, the optimum normalized amplitude $D/\overline{n}$ being from 0.9 to 0.96 and the optimum normalized period ranging from $T/\overline{n} = 0.5$ to 0.7 for multimodal optimization problems and a standard real coded Genetic Algorithm.

In this paper, we have adapted this scheme, defining what we call Standard Saw-tooth Evolutionary Programming (SSEP). Instead of using 5, population size in each generation is calculated from the last generation population size as:

$$n(t) = n(t-1) - N \tag{6}$$

N being the saw-tooth slope and $n(0) = N_{\mathrm{P}} = 1000$ being the number of individuals in the initial population. The value of D is calculated according to the guidelines previously described. For the parameter T, we estimate its value from the maximum number of generations G and a parameter r that defines the maximum number of restarts, $T = \mathrm{int}(G/r)$. The use of this new parameter r is justified, since a better control of the increase in the diversity of the algorithm is achieved by defining the maximum number of restarts (that is, the maximum number of saw-tooth oscillations) than by defining the amplitude T of each saw-tooth. With these two parameters, the saw-tooth slope is obtained as $N = T/2D$. Figure 3(a) is an example of a standard saw-tooth scheme for the Vehicle dataset, including all the parameters mentioned.

4 Saw-Tooth Evolutionary Programming Algorithm Guided by the Variance of Best Individual Distributions

The previously defined scheme has been enhanced considering the variance of the best individual distributions, forcing the beginning of a new saw-tooth oscillation when the best individuals have converged and the performance of the algorithm decreases. Consequently, if the variance of best individual fitnesses has significantly decreased, the population size falls. Otherwise a restart is performed, new individuals being added to the current population. This methodology has been called Guided Saw-tooth Evolutionary Programming (GSEP).

The condition that must be fulfilled in order to force a restart of the population is that the difference of variance of the best individual fitnesses between two consecutive generations decreases by a specific percentage (λ) with respect to the previous generation variance, previously establishing the most appropriate distribution to represent the best individual fitnesses. As a first approximation, the best individual fitness values have been characterized using two distinct distributions, uniform distribution and normal distribution. Considering the uniform distribution, the variance is estimated by:

$$\widehat{\sigma}^2 = \frac{1}{12}\left(f_{max} - f_{min}\right)^2 \tag{7}$$

f_{max} and f_{min} being, respectively, the maximum and minimum fitness value of the best individual set B, and, considering the normal distribution, the variance is estimated as:

$$\widehat{\sigma}^2 = \frac{1}{|B|} \sum_{i=1}^{|B|} \left(f_i - \overline{f}\right)^2 \tag{8}$$

f_i being the fitness of the individual i of the set of best individuals B and $\overline{f}$ the mean fitness value of all individuals in B.

Finally, a minimum number of individuals m_{n} in the population is defined based on the amplitude D of saw-teeth, $m_{\mathrm{n}} = 1000 - 2D$. Restart will also be forced if the population size achieves this minimum. The number of the best individuals whose variance is studied in order to guide the algorithm ($|B|$) is constant during the evolution process and has been fixed to m_{n}. As a result, the number of individuals in each generation is obtained as:

- If $[(\widehat{\sigma}_t^2 < \widehat{\sigma}_{t-1}^2)$ and $(\widehat{\sigma}_{t-1}^2 - \widehat{\sigma}_t^2) < \lambda \widehat{\sigma}_{t-1}^2]$ or $[(n(t-1) - N) \leq m_{\mathrm{n}}]$,
  Perform a restart, generating new individuals until filling initial population size, $n(t) = 1000$.
- Otherwise,
  Decrease population size, $n(t) = n(t-1) - N$, or keep population size to the minimum value if the maximum number of restarts has been achieved, $n(t) = m_{\mathrm{n}}$.

5 Experiments

The proposed GSEP algorithm is applied to six datasets taken from the UCI repository, [10], to test its overall performance as compared to EP and SSEP algorithms. The selected databases include both binary classification problems (German, Heart-statlog, Ionosphere) and multiclass classification problems (Balance, Glass and Vehicle). The experimental design was conducted using a holdout cross-validation procedure, with 30 runs and 70% of instances for the training set and 30% for the generalization set.

Regarding the configuration of the different experiments, the specific parameters for each database have been fine tuned by a trial and error process and summarized in Table 5. Considering an estimated mean population size $\overline{n}$ of 500 and the guidelines proposed in [1], D parameter value should range between 480 and 490. Consequently, we have considered two different values of the m_{n} parameter , 20 and 40, for SSEP and GSEP algorithms. The period T of saw-teeth is determined by the maximum number of restarts r and maximum number of generations G, both specified in Table 5. For GSEP experiments, r value has been increased in one additional restart, in order to avoid an excessive number of generations with a minimum population size.

In order to graphically evaluate the performance of the proposed methodology, the mean and the maximum fitness of the population have been plotted versus the number of generation in Fig. 1 and Fig 2. The depicted values correspond

Table 1. Non-common parameters values

Dataset	Balance	German	Glass	Heart	Ionos	Vehicle
G	300	150	200	100	350	1500
$[m, M_{\mathrm{E}}, M_{\mathrm{I}}]$	[3,4,5]	[2,3,4]	[7,8,9]	[1,1,2]	[3,4,5]	[6,7,8]
r	5	1	2	2	2	4
λ	0.005	0.0075	0.0075	0.0075	0.005	0.001

to the average over 30 runs of Vehicle experiment, using both SSEP and GSEP schemes. The dynamic guided Saw-tooth algorithm achieves an overall better performance (0.6% better final fitness). At the final generations, mean fitness is higher using the GSEP scheme than using the SSEP and, throughout the evolution process, the guided application of the saw-tooth reinitialization leads to better fitness peaks.

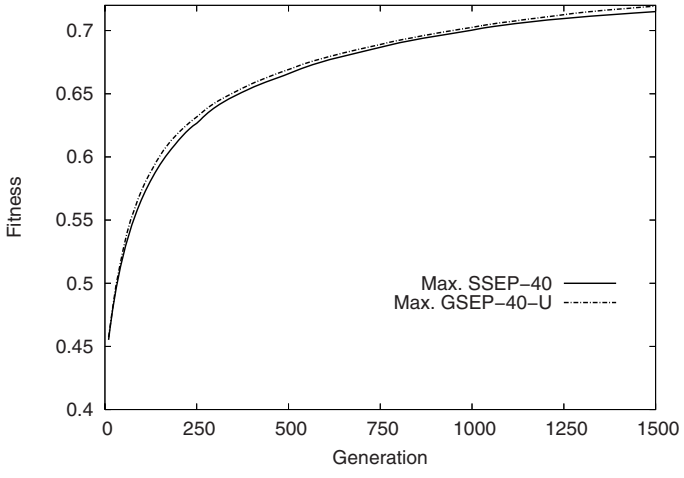


Fig. 1. Maximum fitness average for 30 runs using SSEP-40 and GSEP-40-U algorithms

Table 2 shows the mean value and standard deviation of results of the Correctly Classified Rate (CCR) for training and generalization sets and the corresponding number of net connections in 30 runs of the experiment, for the EP algorithm and the best performing variant of SSEP and GSEP algorithms. As the algorithms returned both the best CCR and cross-entropy individuals, Table 2 only includes the most efficient approach in each given problem. The algorithm that yields better generalization results is represented in bold print. The saw-tooth algorithms (SSEP and GSEP) can be observed to outperform the original EP algorithm in four out of the six problems analyzed. Moreover, the models obtained have a lower number of connections in five out of six databases, which suggests that including periodical reinitializations allows the evolutionary

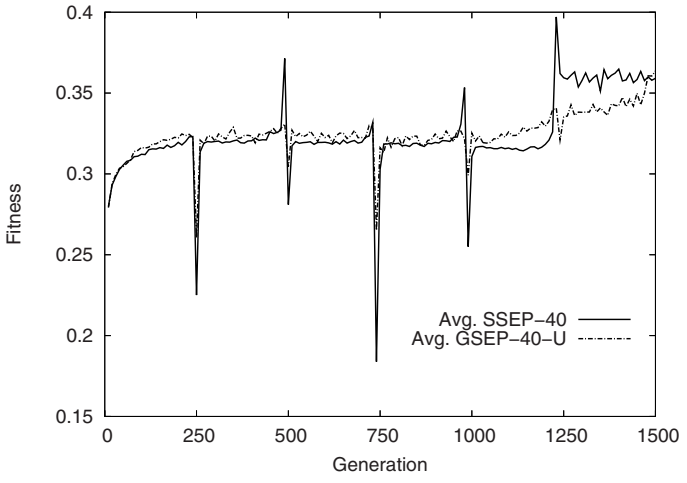
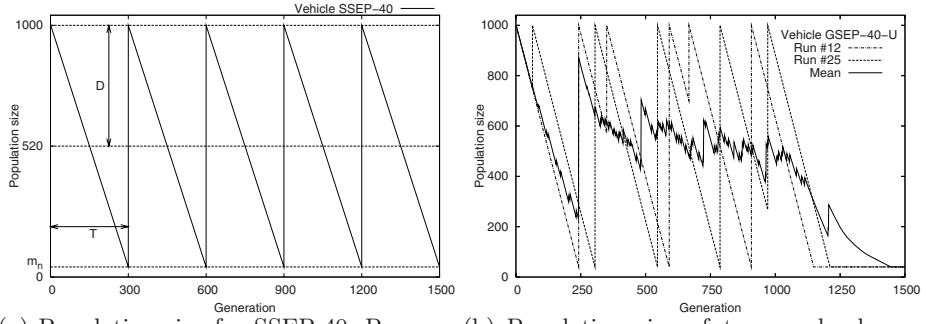


Fig. 2. Mean fitness average for 30 runs using SSEP-40 and GSEP-40-U algorithms

Table 2. Best statistical (mean and standard deviation, SD) CCR results in training and testing sets, number of connections and population size throughout the 30 runs for: Evolutionary Programming with 1000 individuals (EP), Standard Saw-tooth EP (SSEP) and Guided Saw-tooth EP (GSEP), with 20 and 40 individuals as minimum population size and considering uniform (U) and normal (N) distributions

Dataset (Elitism)	Method	CCR_T Mean ± SD	CCR_G Mean ± SD	#connect Mean ± SD	Pop. Size Mean ± SD
Balance	EP	95.47 ± 1.50	**94.10** ± 1.64	30.73 ± 2.16	1000.00
(CCR)	SSEP-40	94.48 ± 1.35	93.18 ± 1.86	**30.43** ± 2.46	524.00
	GSEP-40-U	94.31 ± 1.56	93.65 ± 1.56	30.60 ± 2.25	527.05 ± 45.77
German	EP	78.66 ± 1.48	73.05 ± 1.58	95.20 ± 21.90	1000.00
(Fitness)	SSEP-40	78.48 ± 1.06	72.87 ± 1.77	88.17 ± 21.08	520.50
	GSEP-40-N	78.20 ± 1.44	**73.43** ± 2.22	**85.80** ± 19.56	553.86 ± 46.08
Glass	EP	72.61 ± 3.00	67.23 ± 4.10	77.57 ± 8.01	1000.00
(CCR)	SSEP-40	71.76 ± 2.96	67.99 ± 4.04	**73.63** ± 7.41	544.48
	GSEP-40-N	70.99 ± 2.96	**68.93** ± 4.53	76.37 ± 7.26	389.69 ± 18.70
Heart	EP	86.06 ± 0.90	**86.91** ± 2.06	17.27 ± 2.03	1000.00
(Fitness)	SSEP-20	86.34 ± 1.22	85.44 ± 1.65	**16.47** ± 2.37	520.00
	GSEP-20-N	86.50 ± 0.96	86.37 ± 1.85	17.23 ± 2.57	366.91 ± 35.53
Ionos.	EP	98.27 ± 0.93	92.53 ± 1.80	**73.77** ± 9.91	1000.00
(Fitness)	SSEP-20	98.43 ± 0.83	92.41 ± 2.08	80.83 ± 11.64	510
	GSEP-20-U	98.43 ± 0.75	**92.61** ± 1.80	77.47 ± 12.83	392.20 ± 18.73
Vehicle	EP	79.94 ± 1.45	78.33 ± 2.74	96.47 ± 6.62	1000.00
(Fitness)	SSEP-20	80.12 ± 1.86	78.50 ± 2.93	95.93 ± 6.67	510.00
	GSEP-40-U	80.55 ± 1.84	**78.66** ± 2.15	**95.03** ± 10.05	449.21 ± 47.40



(a) Population size for SSEP-40. Parameter of saw-teeth are the following: $T = 300$, $D = 480$, $m_n = 40$ and $r = 4$

(b) Population size of two randomly selected executions and mean population size average for 30 executions using GSEP-40-U algorithm.

Fig. 3. Population size for SSEP and GSEP schemes

process to explore simpler structures that offer better generalization results. Finally, the GSEP scheme proposed yielded better results than SSEP in all the experiments evaluated. On the other hand, the efficiency of all the algorithms has been estimated by obtaining the mean size of the population throughout all the generations (also presented in Table 2). The values represented correspond to the average and standard deviation of this mean population size during the 30 runs of the different experiments. Both EP and SSEP algorithms have a fix population scheme, their standard deviation being equal to 0. In Fig. 3 average population size over the 30 runs has been plotted for SSEP and GSEP, together with two example runs for GSEP. In general, saw-tooth schemes (SSEP and GSEP) result in a lower computational cost. Moreover, for Glass, Heartlog and Ionosphere experiments, GSEP obtains a lower population size than SSEP.

6 Conclusions

The application of the standard saw-tooth scheme has proved viable in the designing of Evolutionary MLP Neural Networks, providing diversity to the evolutionary process and resulting in a more efficient and accurate algorithm. The proposed GSEP algorithm improves the performance of the SSEP algorithm and could be easily introduced into any existing EA. This proposed scheme has been presented as an enhanced version of the standard saw-tooth scheme, performing the population restart when the difference of variance between two generations is lower than a percentage of previous variance. The evaluation of the algorithm for a wide, thought not exhaustive, range of problems examined showed results that are comparable to those of other classification techniques found in machine learning literature [16]. In this way, it can be affirmed that distribution of the best individual fitnesses can be considered a suitable tool for guiding restarts and introducing diversity into the EP algorithm evaluated. Since the results show that the improvement is consistent in the selected databases, we are extending

experimentation using more databases, in order to apply of statistical tests and analyze in depth the efficiency of the proposed methodologies.

Acknowledgements. This work has been partially subsidized by the TIN 2005-08386-C05-02 project of the Spanish Inter-Ministerial Commission of Science and Tech. (MICYT) and FEDER funds. The research of P.A. Gutiérrez has been backed by the FPU Predoctoral Program (Spanish Ministry of Edu. and Sci.).

References

1. Koumousis, V.K., Katsaras, C.P.: A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. IEEE Transactions on Evolutionary Computation 10(1), 19–28 (2006)
2. Fogel, L.: Artificial Intelligence through Simulated Evolution, 1st edn. John Wiley & Sons, New York (1996)
3. Yao, X.: Evolving artificial neural networks. Proc. of the IEEE 87(9), 1423–1447 (1999)
4. Goldberg, D., Deb, K., Clark, J.: Genetic algorithms, noise, and the sizing of populations. Complex Systems 6(4), 333–362 (1992)
5. Smith, R.: Adaptively resizing populations: An algorithm and analysis. In: Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 653–653 (1993)
6. Cobb, H., Grefenstette, J.: Genetic algorithms for tracking changing environments. In: Proceedings of the 5th International Conf. on Genetic Algorithms, pp. 523–530 (1993)
7. Eshelman, L.: The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In: Proc. Foundations Genetic Algorithms-1, pp. 257–266 (1991)
8. Goldberg, D.: Sizing populations for serial and parallel genetic algorithms. In: Proceedings of the Third International Conference on Genetic Algorithms, pp. 70–79 (1989)
9. Hervás, C., Martínez-Estudillo, F.J., Gutiérrez, P.A.: Classification by means evolutionary product-unit neural networks. In: Proc. of the 2006 International Joint Conference on Neural Networks, Vancouver, Canada, pp. 2834–2842 (2006)
10. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
11. Angeline, P.J., Saunders, G.M., Pollackm, J.P.: An evolutionary algorithm that constructs recurrent neural networks. IEEE Trans. on Neu. Net. 5(1), 54–65 (1994)
12. Yao, X., Liu, Y.: Evolving artificial neural networks through evolutionary programming. In: Fogel, L., Angeline, P., Bäck, T. (eds.) Proc. of the Fifth Annual Conference on Evolutionary Programming V, pp. 257–266 (1996)
13. Martínez-Estudillo, A.C., Hervás-Martínez, C., Martínez-Estudillo, F.J., García-Pedrajas, N.: Hybridization of evolutionary algorithms and local search by means of a clustering method. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 36(3), 534–545 (2006)
14. Martínez-Estudillo, A.C., Martínez-Estudillo, F.J., Hervás-Martínez, C., García-Pedrajas, N.: Evolutionary product unit based neural networks for regression. Neural Networks 19(4), 477–486 (2006)
15. Ventura, S., Romero, C., Zafra, A., Delgado, J., Hervás, C.: JCLEC: a java framework for evolutionary computation. Soft Computing (published online, 2007)
16. Landwehr, N., Hall, M., Frank, E.: Logistic model trees. Mach. Learn. 59(1-2), 161–205 (2005)

Using a Genetic Algorithm for Editing k-Nearest Neighbor Classifiers

R. Gil-Pita[1] and X. Yao[2,3,⋆]

[1] Teoría de la Señal y Comunicaciones, Universidad de Alcalá, Madrid (Spain)
[2] Computer Sciences Department, University of Birmingham, Birmingham (UK)
[3] Nature Inspired Computation and Applications Laboratory, University of Science and Technology of China, Hefei, Anhui 230027 (P.R. China)

Abstract. The edited k-nearest neighbor consists of the application of the k-nearest neighbor classifier with an edited training set, in order to reduce the classification error rate. This edited training set is a subset of the complete training set in which some of the training patterns are excluded. In recent works, genetic algorithms have been successfully applied to generate edited sets. In this paper we propose three improvements of the edited k-nearest neighbor design using genetic algorithms: the use of a mean square error based objective function, the implementation of a clustered crossover, and a fast smart mutation scheme. Results achieved using the breast cancer database and the diabetes database from the UCI machine learning benchmark repository demonstrate the improvement achieved by the joint use of these three proposals.

1 Introduction

Editing a k-nearest neighbor (kNN) consists of the application of the kNN classifier with an edited training set in order to improve the performance of the classifier in terms of error rate [1]. This edited training set is a subset of the complete training set in which some of the training patterns are excluded. So, depending on the characteristics of the database [2], and due to the exclusion of these patterns, the kNN may render better results using the edited set, in terms of both error rate and computational cost.

Genetic algorithms (GA) have been successfully applied to select the training patterns included in the edited training set. In [3] a study of editing kNN classifiers using GAs with different objective functions is presented. Several databases like the Iris database or the Heart database are used in the experiments. The paper concludes that, from the analyzed objective functions, the best results are obtained when the counting estimator with penalizing term is selected as objective function. Other interesting article is [4], in which a GA with a novel crossover method is applied. When two parents are crossed, a high number of

⋆ This work has been partially funded by the Comunidad de Madrid/Universidad de Alcalá (CCG06-UAH/TIC-0378) and the Spanish Ministry of Education and Science (TEC2006-13883-C04-04/TCM).

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1141–1150, 2007.

possible offsprings are evaluated, and the best two individuals are selected. The work presented in [5] is other interesting paper that studies the kNN edited with other heuristic techniques, in which the authors study the use of tabu search to solve the problem of editing a 1NN classifier (nearest neighbor rule). They use the counting estimator objective function with a penalizing term, and they evaluate the results with the iris database and a synthetic two-dimensional database. At last, the use of a multi-objective evolutionary algorithm to simultaneously edit and select the features of an 1NN classifier is evaluated in [6].

In this paper we propose a novel application of GAs for editing kNN classifiers. We describe the design of a genetic algorithm in order to edit training sets for kNN classifiers, focusing on the selection of the objective function to be minimized, and on the different parameters of the genetic algorithm like, for example, the crossover and the mutation techniques. Three improvements are proposed to the editing process using genetic algorithms: the use of a mean square error (MSE) based objective function, the implementation of a clustered crossover, and a fast smart mutation scheme. Results are compared using the breast cancer database and the diabetes database from the UCI machine learning benchmark repository. The computational cost after training and the classification error rate are considered in the study.

2 Materials and Methods

In this section we carry out a brief description of the main classification method this paper deals with: the kNN. After describing the statistical basis of the kNN method, we mathematically describe the editing process of a kNN method, and how the genetic algorithms can be used for editing training sets.

2.1 kNN Statistical Analysis

The kNN classifier is statistically inspired in the estimation of the posterior probability $p(H_i|\mathbf{x})$ of the hypothesis H_i, conditioned to the observation point $\mathbf{x}$. Considering a volume around the observation point that encompasses k patterns of the training set and $k[i]$ patters belonging to hypothesis H_i, then equation (1) is an approach of the posterior probability of the class H_i [7].

$$p(H_i|\mathbf{x}) = \frac{p(\mathbf{x}|H_i)p(H_i)}{p(\mathbf{x})} \simeq \frac{k[i]}{k} \tag{1}$$

The Maximum A Posteriori criterion establishes that, for a given observation $\mathbf{x}$, the decision that maximizes the associated posterior probability must be taken. The kNN method fixes k, the number of patterns included in the volume, being these patterns the k nearest (less distanced) patterns from the observation point. The decision is taken by evaluating the values of $k[i]$, $i = 1, \ldots, C$, and selecting the class which obtains a highest $k[i]$ value, and, therefore, maximizes approximation the posterior probability $p(H_i|\mathbf{x})$ given by equation (1). Concerning the distance measurement, in this paper we use the Euclidean distance, so that the volumes are hyper-spheres around the observation point.

2.2 Editing a Training Set with a GA

The edited training set is defined by the indexes of the patterns included in the subset. In order to apply genetic optimization, each subset is associated to a binary vector $\mathbf{b}$, with N bits, being N the total number of patterns in the original training set, so that if the n-th bit is activated $b[n] = 1$, then the corresponding n-th training pattern is included in the subset. So, being S the number of patterns included in the reduced set so that $S \leq N$, then $\sum_n b[n] = S$. The bit-stream $\mathbf{b}$ is determined by the minimization of a given objective function. In this task, different algorithms can be applied, and we focus on the application of GAs to obtain the value of $\mathbf{b}$.

Concerning the objective function, its selection is an important issue in the design of edited kNN methods. Most of the papers use the classification error as objective function (equation (2)), adding in some cases a penalizing term, to consider the number of patterns in the edited training set.

$$F = \frac{1}{N} \sum_{n=1}^{N} h(\mathbf{x}_n) + \alpha b[n] \tag{2}$$

where $h(\mathbf{x}_n)$ is 1 if the n-th pattern is wrongly classified, and 0 in other cases. This objective function is equivalent to the *counting estimator with penalizing term* (CEPT) function, described in [3].

In this paper GAs are applied to determine the optimal bit-stream $\mathbf{b}$, in order to implement an edited kNN rule. So, the process to determine an edited subset for the kNN method using a GA is described as follows:

- The objective function is evaluated with the validation set for several values of k, and the value that performs better results is selected.
- Using a GA, a subset is selected using the training patterns, trying to minimize the objective function measured with the training set. The genetic algorithm is based on the use of natural selection, local mutations and crossover. The population is composed of $P = 100$ individuals, and the best 10 are selected in each generation. The remaining 90 individuals are then generated by single point crossover of the 10 survivors. Then, binary mutations are applied to the whole population, changing a bit with a probability of 0.8%. This process is iterated $G = 100$ generations.
- During the training process, the objective function over the validation set is calculated for the best individual of each population. The subset that achieves the lowest objective value over the validation set is selected as the final subset.

3 Description of the Proposals

In this section descriptions of the three proposals of this paper are included. The first proposal deals with the definition of a novel objective function, inspired in the mean square error (MSE) function, used in many adaptive systems, like, for

example, neural networks. The second proposal defines a novel crossover strategy for the GA, designed using a priori knowledge of the problem. The third and last proposal establishes a new mutation scheme for the GA, that allows to lead the mutations in the population, improving the performance of the GA in the search of local minima of the objective function.

3.1 Editing kNN with a MSE-Based Objective Function

In this paper we propose the use of a novel objective function, based on the MSE function. Let's consider the kNN as a system with C outputs, so that each output is calculated using equation (1). Therefore, the C outputs of the kNN system are approximations of the posterior probabilities of the data. So, $y_n[i]$, described in equation (3), is the i-th output of the kNN system, obtained for the n-th training pattern $\mathbf{x}_n$.

$$y_n[i] = \frac{k_n[i]}{k} \tag{3}$$

where $k_n[i]$ is the number of nearest patterns of class i for the n-th training pattern. Considering the approximation described in equation (1), the outputs of this kNN system are estimations of the posterior probabilities of the classes. So, the objective function to minimize is designed using all the outputs of the system, by minimizing the mean square error, defined by equation (4).

$$F = \frac{1}{NC} \sum_{n=1}^{N} \sum_{i=1}^{C} (y_n[i] - d_n[i])^2 \tag{4}$$

where $y_n[i]$ is the i-th output of the system for the n-th training pattern, and $d_n[i]$ is the desired i-th output for the n-th training pattern, so that $d_n[i]$ is 1 for the patterns of the i-th class and 0 for the rest. In function of the Kronecker delta, $d_n[i] = \delta[i - c_n]$, being c_n the index of the class for the n-th training pattern. Replacing (3) in (4), we obtain (5).

$$F = \frac{1}{NC} \sum_{n=1}^{N} \sum_{i=1}^{C} \left(\frac{k_n[i]}{k} - \delta[i - c_n] \right)^2 \tag{5}$$

The error surface defined by this function is smoother than those obtained using counting estimator based functions, making easier the obtaining of its local minima.

3.2 Editing kNN Using a GA with Clustered Crossover

Single point crossover (SPC) and random crossover (RC) are two of the most commonly used crossover methods used in the literature. Single point crossover generates the offspring by combining the first part of the bit stream of one parent with the last part of the bit stream of the other parent. On the other hand, random crossover generates the offsprings randomly selecting each bit-gene

from one of the parents. These two schemes do not consider possible relationship between the genes.

In this paper we propose a novel crossover scheme for the GA, denominated clustered crossover (CC), in order to improve the determination of the best subset of the training set, that minimizes the selected objective function. In the problem of the selection of the edited training set, it is possible to determine the relationship between the different bits of the bit stream. Each gene is related to the inclusion in the reduced subset of a training pattern. So, the value of a given gene is related to the performance of the classifier in the region of the space around the associated training pattern. Therefore, genes can be grouped into clusters considering the spatial position of their training patterns, using a non supervised clustering technique. In this paper we apply the k-means clustering algorithm [8] to the training data, so that these patterns are grouped in some sets of near patterns (clusters). Each group of patterns defines a cluster of genes, that is considered as a transfer unity in the crossover process. So, the bit-stream of the offsprings are obtained by mixing the clusters of genes of two parents. The clustering process is carried out every generation, and the number of clusters has been selected at random. So, every generation the crossover is carried out with different gene clusters.

3.3 Editing kNN Using a GA with a Fast Smart Mutation Scheme

In this section we describe the application of a mutation scheme that allows to select the best gene or group of genes to be changed, taking into account the variations of the objective function with respect to each gene for a given edited set. We design a fast method for evaluating the error variation when each gene is changed, and we propose a mutation strategy based on these variations of the objective function. We denominate this mutation scheme as "fast smart mutation" (FSM), as it allows to increase the effectiveness of the mutation stage in the genetic algorithm.

The evaluation of the objective function for all the possible bit mutations of a pattern is implemented taking into account prior knowledge of the objective function. Let's consider the bit stream $\mathbf{b}$, then the goal is to find the bit or bits which changes produce the highest reduction in the performance associated to $\mathbf{b}$.

The change of one bit of $\mathbf{b}$ produces the addition or the removal of a training pattern from the edited subset. It causes changes in the values of $k_n[i]$, with a consequent change in the value of the objective function, that might be considered. Let's consider $B_n(k)$ is the distance from the n-th training pattern to its k-th nearest pattern, then:

- If the m-th bit changes from 0 to 1, then the pattern $\mathbf{x}_m$ must now be considered in the subset. If the distance from this pattern to a training pattern $\mathbf{x}_n$ is lower than $B_n(k)$, then this new pattern replaces the k-th nearest neighbor of the training pattern $\mathbf{x}_n$. Due to the addition of this new pattern of class c_m, the value of $k_n[c_m]$ is incremented in 1, and due to the removal of the k-th training pattern, the value of $k_n[c_n^k]$ is decremented in 1, where c_n^k is the class of the k-th nearest neighbor of the training pattern $\mathbf{x}_n$.

– If the m-th bit changes from 1 to 0, then the pattern $\mathbf{x}_m$ is removed from the subset. If the distance from this pattern to a training pattern $\mathbf{x}_n$ is lower than $B_n(k)$, then this pattern will cause changes in the values of $k_n[i]$. The pattern $\mathbf{x}_m$ will not continue in the group of the k nearest neighbors of the pattern $\mathbf{x}_n$, and there will be a new pattern in this group. Due to the removal of this pattern of class c_m, the value of $k_n[c_m]$ is decremented in 1, and due to the inclusion of the $k+1$-th training pattern in the group, the value of $k_n[c_n^{k+1}]$ is incremented in 1.

Equation (6) represents the function $f_{mn}[i]$, the variations in the values of $k_n[i]$ due to a change in the m-th.

$$f_{mn}[i] = \begin{cases} D_{mn} \cdot (\delta[i - c_m] - \delta[i - c_n^k]), & if\ b[m] = 0 \\ D_{mn} \cdot (\delta[i - c_n^{k+1}] - \delta[i - c_m]), & if\ b[m] = 1 \end{cases} \tag{6}$$

where D_{mn} is 1 if the distance from the pattern $\mathbf{x}_m$ to the pattern $\mathbf{x}_n$ is lower than $B_n(k)$, and 0 in other case. So, the MSE-based objective function F_m obtained after changing the m-th gene is represented in equation (7).

$$F_m = \frac{1}{CN} \sum_{n=1}^{N} \sum_{i=1}^{C} \left(\frac{k_n[i] + f_{mn}[i]}{k} - \delta[i - c_n] \right)^2 \tag{7}$$

The variation in the objective function $\Delta_m = F_m - F$ due to a change in the m-th bit can be expressed using equation (8).

$$\Delta_m = \frac{1}{CNk^2} \sum_{n=1}^{N} -2k f_{mn}[c_n] + \sum_{i=1}^{C} f_{mn}[i]^2 + 2k_n[i] f_{mn}[i] \tag{8}$$

Using (6) in (8), we obtain (9).

$$\Delta_m = \frac{2}{k^2 CN} \sum_{n=1}^{N} D_{mn}(1 + g_{mn}) \tag{9}$$

where g_{mn} is defined by equation (10).

$$g_{mn} = \begin{cases} -1, & if\ b[m] = 0\ \ and\ \ c_n^k = c_m \\ h_{0mn}, & if\ b[m] = 0\ \ and\ \ c_n^k \neq c_m \\ -1, & if\ b[m] = 1\ \ and\ \ c_n^{k+1} = c_m \\ h_{1mn}, & if\ b[m] = 1\ \ and\ \ c_n^{k+1} \neq c_m \end{cases} \tag{10}$$

being h_{0mn} and h_{1mn} defined by equations (11) and (12), respectively.

$$h_{0mn} = \begin{cases} k_n[c_m] - k_n[c_n^k] + k, & if\ c_n = c_n^k \\ k_n[c_m] - k_n[c_n^k] - k, & if\ c_n = c_m \\ k_n[c_m] - k_n[c_n^k], & other\ case \end{cases} \tag{11}$$

$$h_{1mn} = \begin{cases} -k_n[c_m] + k_n[c_n^{k+1}] - k, & if\ c_n = c_n^{k+1} \\ -k_n[c_m] + k_n[c_n^{k+1}] + k, & if\ c_n = c_m \\ -k_n[c_m] + k_n[c_n^{k+1}], & other\ case \end{cases} \tag{12}$$

The value of Δ_m (equation (9)) is evaluated for all the possible values of m, in each generation and for every individual. The described algorithm allows to quickly evaluate the variation of the objective function with a unique bit change. So, the change in the value of m that efforts the lowest Δ_m will cause the highest reduction of the objective function.

The GA can be speeded up changing more than one bit in every mutation. In many classification environments, the large size of the training set makes this method quite slow, in so only a gene is changed for each individual every mutation stage. On the other hand, using the clustering process described in subsection 3.2, it is possible to establish groups of "independent" genes. A change in a bit that belongs to a cluster affects to the performance of the classifier in the region of the space nearer to the corresponding training pattern. So, we propose to use the gene clusters to select a group of genes to be mutated. For each cluster, the value of m that efforts the lowest Δ_m is changed, which allows to mutate as many genes as clusters.

The implementation of the algorithm requires the previous calculation of the values of $k_n[i]$, c_n^k and c_n^{k+1}. The process of the genetic algorithm with fast smart mutation is described as follows:

1. The initial population with 100 individuals is generated, all the variables are initialized.
2. The mean square error is measured for every individual of the population. The values of Δ_m are obtained.
3. The k-means algorithm is applied to the training set. The number of clusters is selected at random.
4. For each cluster and each individual, the gene with the value of m that efforts the lowest value of Δ_m is muted.
5. Every 10 generations, clustered crossover is applied to the data. 10 best individuals are chosen as parents, and remaining 90 individuals are generated by clustered crossover of the parents.
6. The validation error of the best individual is calculated.
7. The process is iterated in step 2, until 100 generations are reached.
8. Finally, the selected individual is the one that achieved the lowest validation error.

4 Results

This section includes the results obtained by the methods described in the paper. The databases used in the experiments of the paper have been the breast cancer database and the diabetes database, collected from the UCI machine learning benchmark repository. Choosing these two databases we try to be able to compare the performance of the different methods in two different environments, allowing to extract more general conclusions.

In order to carry out the experiments, each database has been divided in three subsets: the training set, the validation set and the test set. The training

set has been used to generate the edited subsets. The validation set has been used to select the best classifier, and to determine the values of k. The test set has been used to evaluate the final error rate for each classifier. This third set has not been used during the design of the classifier. These three databases and the data preparation techniques are identical to those used in other papers [9,10], allowing to make comparisons of the obtained results with other different type of classifiers. Table 1 shows a summary of the main characteristics of the used two databases.

Table 1. Characteristics of the databases

	Breast cancer	Diabetes
Number of classes C	2	2
Number of inputs L	9	8
Total number of patterns	699	768
Number of training patterns N	349	384
Number of validation patterns	175	192
Number of test patterns	175	192
k value (using the validation set)	3	27

The parameter k of the kNN method is a user-specific parameter. In this work we have selected it in a first stage, making use of the validation set, and it has remained fixed for the rest of the experiments. Different kNN classifiers with values of k from 1 to 50 have been implemented using each database, and the value of k that efforts the lowest classification error rate over the validation set has been selected. This value has been $k = 3$ for the breast cancer database and $k = 27$ for the diabetes database.

In order to assess the performance of the classification methods, the error rate over the test set is used. Due to the small size of the test sets, the precision in the estimation of the error rate is considerably low, and some statistical analysis of the results must be used. So, each experiment has been repeated 30 times, measuring the error rate for each experiment. Results are represented in function of the mean, the standard deviation, the maximum, the minimum of the error rate over the 30 experiments. The average number of patterns selected in the edited subset (S) is also included. Table 2 shows the results obtained by the different methods for the breast cancer database and the diabetes database.

From the obtained results, we can derive the next conclusions:

- The use of the proposed MSE-based objective function has an associated reduction greater than 12%, when it is compared to the use of the CEPT objective function [3].
- The use of the proposed clustered crossover doest not significantly improve the performance in the case of the diabetes database, but it achieves a reduction of 15% in the error rate in the case of the breast cancer database.

– The results obtained by the joint use of the three proposals has an associated reduction greater than 10%, compared to the use of a kNN classifier without editing technique. Obtained results demonstrate the good accuracy of the proposed GA-based editing technique.

Table 2. Results (%) obtained for the different methods studied in the paper

Editing	*Breast Cancer*					*Diabetes*				
technique	*Mean*	*Std*	*Max*	*Min*	*S*	*Mean*	*Std*	*Max*	*Min*	*S*
none	1.14	0.00	1.14	1.14	349	21.88	0.00	21.88	21.88	384
Wilson [1]	1.71	0.00	1.71	1.71	323	27.08	0.00	27.08	27.08	262
GA CEPT SPC [3]	1.96	1.06	4.57	0.00	101	22.76	2.00	26.04	18.75	173
GA MSE SPC	1.43	0.76	2.86	0.00	157	19.84	1.27	21.88	16.67	192
GA MSE RC	1.68	0.78	4.00	0.57	163	19.62	1.18	21.88	17.19	193
GA MSE CC	1.22	0.65	3.43	0.00	186	19.60	1.00	22.40	18.23	191
GA MSE CC FSM	0.72	0.54	2.29	0.00	174	19.39	1.63	22.92	16.67	195

5 Conclusions

In this paper genetic algorithms have been successfully applied to select the training patterns included in an edited set of a kNN classifier. We have proposed three improvements of the editing process using genetic algorithms. Considering the statistical properties of the kNN classifier, we have proposed a novel mean square error based objective function, which performs better than the counting estimator based objective function. The second proposal presents an analysis of the relationship of the genes in the GA, which is used to propose a clustered crossover. At last, a new fast smart mutation scheme that allows to quickly evaluate the variations in the MSE-based objective function for a change in one bit is described.

Results achieved using the breast cancer database and the diabetes database from the UCI machine learning benchmark repository have been included. The obtained results make the joint use of the three proposed methods quite interesting. Comparing these results with the best one obtained using kNN without editing, with Wilson's editing, and with GA-based editing using CEPT and SPC, the proposed method achieves an average reduction of greater than 10% for the considered databases (36% for the breast cancer database and 12% for the biabetes database).

References

1. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited datasets. IEEE Transactions on Systems, Man and Cybernetics 2, 408–421 (1972)
2. Mollineda, R.A., Sánchez, J.S., Sotoca, J.M.: Data characterization for effective prototype selection. In: Marques, J.S., Pérez de la Blanca, N., Pina, P. (eds.) IbPRIA 2005. LNCS, vol. 3523, pp. 27–34. Springer, Heidelberg (2005)

3. Kuncheva, L.I.: Fitness functions in editing k-NN reference set by genetic algorithms. Pattern Recognition 30(6), 1041–1049 (1997)
4. Ho, S.Y., Liu, C.C., Liu, S.: Design of an optimal nearest neigbor classifier using an intelligent genetic algorithm. Pattern Recognition Letters 23, 1495–1503 (2002)
5. Cerverón, V., Ferri, F.J.: Another move toward the minimum subset: a tabu search approach to the condensed nearest neighbor rule. IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics 31(3), 408–413 (2001)
6. Chen, J.-H., Chen, H.-M., Ho, S.-Y: Design of Nearest Neighbor Classifiers Using an Intelligent Multi-objective Evolutionary Algorithm. In: Zhang, C., W. Guesgen, H., Yeap, W.-K. (eds.) PRICAI 2004. LNCS (LNAI), vol. 3157, pp. 262–271. Springer, Heidelberg (2004)
7. Bishop, C.M.: Neural networks for pattern recognition. Oxford University Press Inc, New York (1995)
8. Hartigan, J.A., Wong, M.A.: Algorithm AS 163: A k-means clustering algorithm. Applied Statistics 28(1), 100–108 (1979)
9. Yao, X., Liu, Y.: A new evolutionary system for evolving artificial neural networks. IEEE Transactions on Neural Networks 8(3), 694–713 (1997)
10. Islam, M.M., Yao, X., Murase, K.: A constructive algorithm for training cooperative neural network ensembles. IEEE Trainsactions on Neural Networks 14(4), 820–834 (2003)

An Evolution of Geometric Structures Algorithm for the Automatic Classification of HRR Radar Targets

Leopoldo Carro-Calvo, Sancho Salcedo-Sanz, Roberto Gil-Pita, Antonio Portilla-Figueras, and Manuel Rosa-Zurera

Department of Signal Theory and Communications, Universidad de Alcalá
28871 Alcalá de Henares, Madrid, Spain
sancho.salcedo@uah.es

Abstract. This paper presents a novel approach to solve multiclass classification problems using pure evolutionary techniques. The proposed approach is called Evolution of Geometric Structures algorithm, and consists in the evolution of several geometric structures such as hypercubes, hyperspheres, hyperoctahedrons, etc. to obtain a first division of the samples space, which will be re-evolved in a second step in order to solve samples belonging to two or more structures. We have applied the EGS algorithm to a well known multiclass classification problem, where our approach will be compared with several existing classification algorithms.

1 Introduction

Genetic algorithms have been widely used for solving very different problems, the majority of times related to optimization. In this field the research work has been massive in the last years and powerful algorithms based on evolutionary computation have been developed.

However, there are other fields in artificial intelligence in which the genetic approach has not been so successful. In classification problems for example, the different evolutionary computing techniques have always played a secondary role: training of neural networks [2]-[7], or generation of fuzzy rules [8], but it is difficult to find a pure evolutionary technique applied to the complete resolution of the problem. The Genetic Programming technique [9] has been applied to some classification problems with some success, though its results in multiclass classification have not been so promising.

The idea behind this paper is to propose a pure evolutionary technique to tackle multiclass classification problems. We have called this technique "Evolution of Geometric Structures Algorithm" (EGS), since it is based on evolving a set of geometric structures (hypercubes, hyperspheres, hyperoctahedrons, etc) to cover the samples space, and then using another evolutionary algorithm to combine them into a single classifier. As will be shown, the idea is to run a genetic algorithm encoding a set of geometric structures for each class, in such a way that the position and size of the geometric structure is evolved. A fitness

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1151–1159, 2007.

function counting the number of correct classification of samples for each class is used to guide the search. In a second step, another evolutionary algorithm is used to decide the classification of the samples within two or more geometric structures. As can be seen, only evolutionary techniques are used to solve the problem. Also, the EGS algorithm is adequate to be implemented in a parallel way, since several genetic populations must be run in the first step of the algorithm.

In this paper we apply the presented technique to the automatic classification of high range resolution radar (HRR) targets, which is a hard problem of classification solved previously in the literature [11]. This kind of radar uses broad-band linear frequency modulation or step frequency waveforms to measure range profiles (signatures) of targets [11], [12]. HRR radar profiles are essentially one-dimensional images of radar targets. A range profile is defined as the absolute magnitude of the coherent complex radar returns, and all phase information is usually discarded. If a range profile is measured with sufficient resolution, the parts of the aircraft that strongly reflect the radar energy, are resolved. Therefore, range profiles provide information about the geometry and structure of the aircraft, and so they are suitable features for automatic aircraft classification. Figure 1 shows an example of the generation of a HRR signal associated to a target in two different orientations. The number of samples with information is related to the maximum size of the target. Data sets of HRR radar profiles can be generated by recording measurements of each target over the values of azimuth and elevation considered.

In the literature, this task is called "Automatic Target Recognition" (ATR) [12]. The ATR problem can be formulated as a multiclass classification problem. The objective is to be able to classify an arbitrary target, based on knowledge derived from previous samples. The Probability Density Functions (PDFs) of the

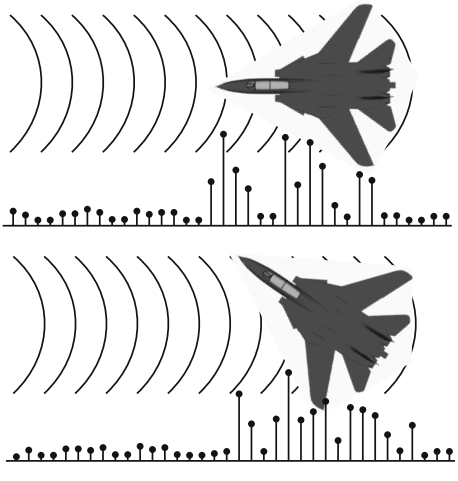


Fig. 1. HRR radar signal example for two different orientations of the target

classes are usually unknown, and only a finite set of well defined cases (training set) is available. Several techniques based on different approaches to the problem and its solution have been developed, all of them having some disadvantages either in terms of accuracy or complexity.

The structure of the rest of the paper is the following: next Section presents the EGS algorithm, describing its different steps and possible modifications for a better performance. Section 3 shows an example of application of the EGS to the ATR problem. Section 4 closes the paper with some final remarks.

2 An Evolution of Geometric Structures Algorithm for Classification

Definitions

1. *Hyper-structures:* the set of points $x \in S$ (S features space of dimension w) such that

$$a_i \cdot p_i \cdot |c_i - x_i| \leq R, \ \forall \ i, \tag{1}$$

this equation defines a hypercube.

$$\sum_{i=1}^{w} a_i \cdot p_i \cdot (c_i - x_i)^2 \leq R, \tag{2}$$

this equation defines a hypersphere.

$$\sum_{i=1}^{w} a_i \cdot p_i \cdot |c_i - x_i| \leq R, \tag{3}$$

this equation defines a hyperoctahedron.

In all the structures presented, the parameters c_i, $i = 1, \cdots, w$ stands for the coordinates of the center of the geometric structure, p_i, $i = 1, \cdots, w$ is a positive scale factor and $a_i \in \{0, 1\}$, $i = 1, \cdots, w$, are binary numbers which reduce the dimensionality of the geometric structure.

2. *Genetic Individual:* A set of N hyperstructures. Each hyperstructure is defined by the equations shown above, with parameters R, $\mathbf{c}$, $\mathbf{a}$ and $\mathbf{p}$.

3. *Genetic Population:* A set of M genetic individuals focussed in a given class C. An point x is decided to belong to class C if this point is inside the structure defined for this class.

With these definitions, the EGS algorithm is constructed in two different steps:

Step 1

In this step, C genetic algorithms (one per class) are launched. Each genetic algorithm is formed by M individuals representing geometric structures in the way defined above. The fitness function of the genetic algorithm is defined as:

$$f(x) = \nu_1 \cdot A + \nu_2 \cdot B \tag{4}$$

where A stands for the percentage of correct answers when classifying a point x as belonging to the corresponding class of the geometric structure, and B is the percentage of correct answers when classifying a point x as not belonging to the corresponding class of the geometric structure. Parameters ν_1 and ν_2 weight the importance of classifying correctly the points belonging and not belonging to the class of the geometric structure. It is important to note that we are evolving a number $N * C$ of structures (N structures per class), but, at this stage, each class is evolved in a different genetic population.

Evolutionary Algorithm dynamics: Once each individual of the population has assigned its fitness value, the mean value for the population is calculated. The individuals with associated fitness values over this value are maintained, whereas the individuals with fitness values under the mean are substituted by new individuals obtained from the crossover of maintained individuals.

The crossover operation is implemented in the following way: Starting from two parents P_1 and P_2, a single offspring individual will be generated. To do this, for each of the N geometric structures encoded in the individuals we can do the following actions:

- Copy the structure of P_1, with probability 0.25.
- Copy the structure of P_2, with probability 0.25.
- Form a new individual as follows: the new radio is the mean value of the P_1 and P_2 radios. The same operation is carried out to obtain the centers and scale factors of the individual. Regarding the parameter a, it is calculated using the logical operation "or" between vectors $\mathbf{a}$ of P_1 and P_2, with a probability of 0.5, and the logical operation "and" with the same probability.

The mutation operator is applied to all the individuals of the population after the crossover operator, with a low probability:

- The radio is modified using a small noise from a uniform distribution $[-0.1, 0.1]$. Choosing randomly a direction, the center is moved towards this direction using a uniform noise in the range $[-1.0, 1.0]$.
- The scale factor of each geometric structure is modified adding an noise from a uniform distribution $[-1.0, 1.0]$.
- Finally, the vector $\mathbf{a}$ is modified applying a flip operation to some of its components.

Step 2

After evolving the set of geometric structures for each class, we have to merge them into a single classifier. This is problematic, since structures may overlap, and thus a sample may be into structures belonging to different classes. In this second step we use a genetic algorithm for assigning a real weight (ω) to each structure, in such a way that the number of correct classifications to be maximum (we consider the real numbers in the interval $\omega \in [-1, 1]$). A voting scheme is carried out, and the samples belonging to two or more geometric structures are assigned to the structure with largest weight (if a sample belongs to two structures of the same class, their values are added). As mentioned, the fitness value of each individual is the total number of correct classifications.

This process can be done in a soft way, by defining variable weights, in such a way that the value of the weight is ω in the center of the hyperstructure, and it has a lower value near the border of the hyperstructure. To do this we define the following weight:

$$\omega^* = (\omega - d \cdot Q) \cdot h(\mathbf{x}), \tag{5}$$

where Q is a value in $(0, 1)$, d is the distance from the center of the geometric structure and $h(\mathbf{x})$ is a binary value which indicates if the point $\mathbf{x}$ is inside the structure ($h(\mathbf{x}) = 1$) or out of the structure ($h(\mathbf{x}) = 0$). An example in a two-dimensional structure (octahedron) is given in Figure 2.

If we use this soft version of the weights, the genetic algorithm must search for values ω and Q for each structure, instead of only for ω as in the former case.

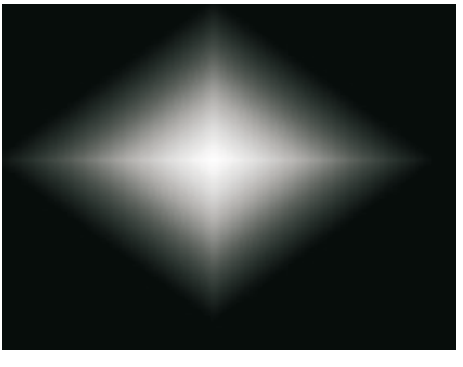


Fig. 2. Example of the soft version in the case of samples shared by two or more structures. In this case an octahedron with a different value in its weight (linear variation following Equation (5)) is considered. The value of the weight is related with the intensity of the hyperstructure: the closer the sample is to the center of the hyperoctahedron, the largest is the associated weight.

3 Experiments and Results

As we mentioned before, we test the propose EGS algorithm in the ATR problem. The term ATR "Automatic Target Recognition" was originated in the early

1980s within the Low Altitude Navigation and Targeting Infra-Red for Night (LANTIRN) program belonging to the U.S. Defense Agency. One of its objectives was to develop a system capable of distinguishing tanks from trucks, jeeps, and other less important targets. In 1988, the United States Defense Advanced Research Projects Agency (DARPA) conducted a study of neural networks and selected ATR as one of the four areas for which application of neural network technology was to be evaluated [13].

A database containing HRR radar profiles of six types of aircrafts (six classes) is used. These signals have been generated using a ray tracing algorithm, and they are the same signals used to generate the results presented in [14]. The assumed target position is head-on with an azimuth range of $\pm 25^o$ and elevations of -20^o to 0^o in one degree increments totaling 1071 radar profiles per class. The length of each profile is 128. These profiles have been aligned using the position of the maximum sample in each vector, and they have been energy normalized and Box-Cox transformed with $\alpha = 0.65$. Once the profiles have been aligned, most of the information is concentred in the central part of the vector, so only $L = 53$ central samples are selected to classify each pattern. Figure 3 shows radar profiles of two different targets, illustrating the magnitude of variation of the radar signature caused by changes in orientation.

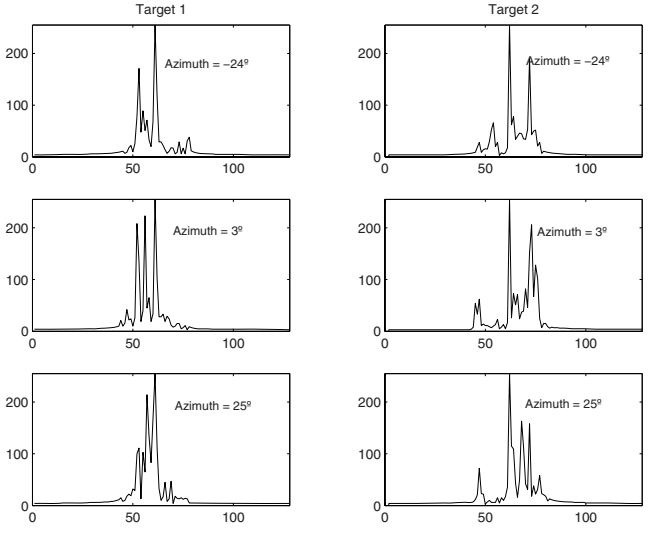


Fig. 3. Radar signatures of two different targets for an elevation angle of -20^o and different azimuth angles

The performance of the classifier will be specified by both the error rate and the computational cost after training. The error rate is defined as the percentage of overall classification errors, and the computational cost after training is defined as the average number of simple operations (sums, products, comparisons, etc) needed to classify each test pattern.

For each experiment carried out, two subsets are used: a training set composed of 960 profiles (160 per class), randomly selected from the original data set (the poses could be different for different targets), and a test set, composed of 1710 profiles. The test set is used to assess the classifier's quality after training. Both sets of data correspond to a value of Signal to Noise Ratio of 20 dB. Recall that the SNR is defined using the peak energy of the signal, given by equation

$$SNR(dB) = 10 \log \left(\frac{\max\{|\mathbf{x}|\}^2}{\sigma^2} \right) \tag{6}$$

The proposed algorithm is going to be tested in this problem, using three different geometric structures, hypercubes, hyperspheres and hyperoctahedrons. We will compare the results obtained with the results of several different algorithms applied to the problem: Diagonal Linear Discriminant Analysis (DLDA), Diagonal Quadratic Discriminant Analysis (DQDA), 1-nearest neighbor (1NN), 3-nearest neighbor (3NN),5-nearest neighbor (5NN), a Multi Layer Perceptron with gradient descent momentum and adaptive learning rate (MLP) and finally a RBF network (RBFN).

Table 1. Comparison of the results (error rate) obtained by the EGS algorithm and other classification algorithms in the ATR problem tackled

Algorithm	DLDA	DQDA	1NN	2NN	3NN	MLP	RBF	EGS-HC	EGS-HS	EGS-HO
Error rate	24.62%	21.11 %	21.99%	19.53%	19.42%	19.88%	14.97%	31.77%	27.12%	17.87%

Table 2. Computational complexity of the algorithms in terms of the number of basic operations needed to classify a single pattern

Algorithm	DLDA	DQDA	1NN	2NN	3NN	MLP	RBF	EGS-HC	EGS-HS	EGS-HO
# of Operations	953	1271	152645	153604	154562	10959	110085	4229	3457	4230

Tables 1 and 2 sumarize the results obtained by the EGS algorithm proposed, with different geometric structures implemented, and the comparison with several other classification techniques in the tackled ATR problem. The first result that can be extracted from the tables refers to the behavior of the different type of structures involved in the EGS algorithm. Note that the hyperoctahedron structure is the one which obtains a best result in terms of error rate of classification. Other structures such as hyperspheres or hyperplanes do not provide such a good results as the hyperoctahedrons, as can be seen in Table 1. All the tested EGS algorithms obtained a similar computational cost, measured as the number of basic operation needed to classify a single pattern.

Regarding the comparison with other techniques, it is interesting to check that the EGS algorithm using hyperoctahedrons (EGS-HO) improves the results of the compared classification methods. In the case of discriminant analysis algorithms, our EGS-HO approach obtains better error rates than DLDA and DQDA

algorithms (24.62% and 21.11% versus 17.87% obtained with the EGS-HO). The computational cost of the DLDA and DQDA approaches, is, on the other hand, slightly smaller than the EGS-HO's: 953 and 1271 operations versus 4230 of the EGS-HO. Our EGS-HO algorithm also improves the results of the nearest neighbors classifiers tested, both in error rate and in computational cost of the algorithm. The MLP approach obtains a good error rate, comparable with those from the NN algorithms, but worse than the EGS-HO approach. The computational cost of this algorithm is also worse than the EGS-HO (10959 operations versus 4230 with the EGS-HO). Finally, the RBF network is the only compared approach which have obtained a slightly better result than the EGS-HO in terms of error rate in classification (14.97% versus 17.87% with the EGS-HO), on the other hand, the RBF network shows a much worse computational cost (110085 operations versus 4230 of the EGS-HO).

4 Conclusions

In this paper we have presented a novel evolutionary-type algorithm for multiclass classification problems, called Evolution of Geometric Structures algorithm. We have analyzed the main characteristics of the algorithm and applied it to the resolution of a well known multiclass classification problem, the Automatic classification of High Range Resolution radar targets, ATR problem. We have shown that our approach is able to obtain results which improves the results of other different classification techniques such as discriminant analysis, k-neares neightbours or a multilayer perceptron. As a final conclusion, the proposed method presents an interesting trade off between error rate and computational cost after training, which might be very beneficial in those applications in which embedded real time implementation is required.

References

1. Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading, MA (1989)
2. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)
3. Whitehead, B.A., Chaote, T.D.: Cooperative-competitive genetic evolution of radial basis function centers and width for time series prediction. IEEE Trans. on Neural Networks 7(4), 869–880 (1996)
4. Rivas, V.M., Merelo, J.J., Castillo, P.A., Arenas, M.G., Castellano, J.G.: Evolving RBF neural networks for time-series forecasting with EvRBF. Information Sciences 165(3-4), 207–220 (2004)
5. Kilmek, M.M., Sick, B.: Architecture optimization of radial basis function networks with a combination of hard- and soft-computing techniques. In: Proc. of the IEEE Systems, Man and Cybernetics International Conference, vol. 5(5-8), pp. 4664–4671 (2003)
6. Yu, H., Zhu, H., Shi, Y.: RBF networks trained by genetic algorithm applied in active control of noise and vibration. Acoustical Science and Technology 25(1), 109–111 (2004)

7. Mak, M.W., Cho, K.W.: Genetic evolution of radial basis function centers for pattern classification. In: Proc. of the IEEE World Conference on Neural Networks, vol. 1, pp. 669–673 (1998)
8. Angelov, P.P., Buswell, R.A.: Automatic generation of fuzzy rule-based models from data by genetic algorithms. Information Sciences 150(1-2), 17–31 (2003)
9. Koza, J.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
10. Salcedo-Sanz, S., Fernández-Villacañas, J.L., Segovia-Vargas, M.J., Bousoño-Calzón, C.: Genetic programming for the prediction of insolvency in non-life insurance companies. Computers & Operations Research 32, 749–765 (2005)
11. Smith, C.R., Goggans, P.M.: Radar Target Identification. IEEE Antennas and Propagation Magazine 35(2), 27–37 (1993)
12. Jacobs, S.P., O'Sullivan, J.A.: Automatic Target Recognition Using Sequences of High Resolution Radar Range-Profiles. IEEE Transactions on Aerospace and Electronic Systems 36(2), 364–381 (2000)
13. DARPA: Neural Network Study. Fairfax, VA: AFCEA International Press (1988)
14. Nelson, D.E., Starzyk, J.A., Ensley, D.D.: Iterated wavelet transformation and signal discrimination for HRR radar target recognition. IEEE Transactions on Systems, Man and Cybernetics 33(1), 52–57 (2003)

Hybrid Cross-Entropy Method/Hopfield Neural Network for Combinatorial Optimization Problems

Emilio G. Ortiz-García and Ángel M. Pérez-Bellido

Department of Signal Theory and Communications, Universidad de Alcalá, 28871 Alcalá de Henares, Madrid, Spain

Abstract. This paper presents a novel hybrid algorithm for combinatorial optimization problems based on mixing the cross-entropy (CE) method and a Hopfield neural network. The algorithm uses the CE method as a global search procedure, whereas the Hopfield network is used to solve the constraints associated to the problems. We have shown the validity of our approach in several instance of the generalized frequency assignment problem.

1 Introduction

The hybridization of heuristic algorithms with local search methods has attracted the attention of many researchers in the last few years [1]-[3]. The interest in these methods is mainly due to the fact that they allow a robust solving of a huge number of difficult problems, in which finding a solution is very hard (impossible in some cases) using traditional optimization methods.

In the literature different ways of hybridizing algorithms can be found. In every hybrid algorithm two sub-procedures coexist: a global search procedure, which is the fundamental part of the algorithm, and a local search one which complements the global search. A classification of hybrid algorithms can be done, depending on the role played for the local search procedure, that includes hybrid [1] and memetic [3] algorithms. The first ones include local search procedures in order to solve the problem constraints. In this way, the global search works only with feasible solutions. However, in the memetic approach, the local search procedure is used to improve the performance of the global search. In both methods, the hybridization of the local procedures can be in Lamarckian mode or in Baldwin mode. In Lamarckian mode, the resulting individual of the local search replaces to the previous individual in the global search. In Baldwinian mode, the individual is not replaced in the global search, but only the objective function value of the solution obtaining after the local search procedure is considered in the global search.

In this paper we focus our work in hybrid algorithms (which use the local search to solve constraints problem). Specifically, we study a new hybrid algorithm which arises of the hybridization of a global search procedure based in the Cross-Entropy method (CE) [5] and a binary Hopfield neural network (HNN)

H. Yin et al. (Eds.): IDEAL 2007, LNCS 4881, pp. 1160–1169, 2007.

[6] as local search procedure. Firstly, we carry out an introduction on the main characteristics of two methods mentioned, and later on, we describe how the hybrid algorithm we propose is organized. Finally, we prove that this hybrid algorithm improves the results obtained with a genetic algorithms as global search procedure in a well known combinatorial optimization problem with constraints.

The rest of the paper is structured as follows: Section 2.1 presents the binary Hopfield neural network used in our approach. Section 2.2 presents a introduction to the Cross-Entropy method including a brief summary of its origin and a discussion of the algorithm in combinatorial optimization problems with binary encoding. Section 3 presents the main characteristics of the HNN-CE algorithm proposed, which is applied to solve the Generalized Frequency Assignation problem in Section 4. Finally, the section 5 finishes the paper giving some conclusions.

2 Background

2.1 The Binary Hopfield Neural Network

In order to satisfy the COP's constraints of a combinatorial optimization problem with binary encoding, we use a binary Hopfield neural network (HNN). The HNN used belongs to a class of binary HNNs [8], [9] where the neurons can only take values 1 or 0.

The structure of the HNN can be described as a graph, where the set of nodes are the neurons, and the set of edges define the connections between the neurons. We map a neuron to an element in the solution $\mathbf{X}$. In order to simplify our notations, we shall use $\mathbf{X}$ to denote the neurons in the HNN. The HNN dynamics can then be described as follows: After a random initialization of every neuron with binary values, the HNN operates in serial mode. This means that only one neuron is updated at a time. Denoting by $x_i(t)$ the state of a neuron on time t, the updating rule is described by

$$x_i(n+1) = \phi(u_i(n)) \tag{1}$$

where ϕ is the updating rule, and $u_i(n)$ is the input equation of the neuron i at time n, which depends on the output equations of the rest of neurons through the COP's constraints.

In this updating rule, neurons x_{ik} are updated in their natural order, i.e., $i = 1, 2, \ldots, n$, $k = 1, 2, \ldots, m$. A modification of this rule can be introduced by performing the updating in a random ordering of the rows (variable i). This way the variability in the feasible solution found is increased. Let $\pi(i)$ be a random permutation of $i = 1, 2, \ldots, n$. The new updating rule of the HNN will be

$$x_i(n+1) = \phi(u_{\pi(i)}(n)) \tag{2}$$

We can define a *cycle* as the update of all neurons in a certain order. In a *cycle* each neuron is updating in the order given by π. After two cycles, we consider that the HNN has converged (in that moment no neuron will change its state). In this way, the final HNN state is a potential solution of the optimization problem considered which fulfills the COP's constraints.

2.2 The Cross-Entropy Method for Optimization

The CE method is an elegant optimization method which comes from the simulation of rare events field [4], [5]. The idea is that the simple algorithm of cross-entropy used for estimating probabilities of rare events could be adapted to solve difficult problems of optimization as well. This adaptation is done by translating the "deterministic" optimization framework to a "stochastic" one, and then using rare event simulation techniques similar to the ones described in [7].

The CE method can be split in two phases:

- Generate a random data sample for the problem, according to a specified mechanism.
- Update the parameters of the method, mainly the probabilities associated to the variables of the solution.

The formal description of the CE for optimization can be done following [7]: Consider an optimization problem consisting in finding the solution (*state*) $\mathbf{x}$ which maximizes a given real-valued, objective function S. Mathematically:

find $\mathbf{x}$ such that,

$$\max_{\mathbf{x}\in\chi} S(\mathbf{x}). \tag{3}$$

where χ is a finite set of states (search space).

The CE method associates an *estimation problem* with the optimization problem described by (3). To do this, a collection of indicator functions $\{I_{S(\mathbf{x})\geq\gamma}\}$ on χ is defined, for various levels $\gamma \in \mathbb{R}$. Then, let $\{f(.;\mathbf{v})\}$ be a family of discrete pdfs on χ, parameterized by a real-valued vector $\mathbf{v} \in [0,1]^n$. We can associate then the optimization problem defined in (3) with the problem of estimating the number

$$l(\gamma) = \mathbb{P}_{\mathbf{v}}(S(X) \geq \gamma) = \sum_{\mathbf{x}} I_{\{S(\mathbf{x})\geq\gamma\}} f(\mathbf{x};\mathbf{v}) = \mathbb{E}_{\mathbf{v}} I_{\{S(\mathbf{x})\geq\gamma\}} \tag{4}$$

where $\mathbb{P}_{\mathbf{v}}$ is the probability measure under which the random state X has pdf $f(.;\mathbf{v})$, and $\mathbb{E}_{\mathbf{v}}$ stands for the expectation operator. This problem is usually called *associated stochastic problem* [7].

Following [7], it can be shown that to estimate $l(\gamma)$, the optimal value for $\mathbf{v}$, so called $\mathbf{v}^*$, can be computed as:

$$\mathbf{v}^* = arg\max_{v} \mathbb{E}_{\mathbf{u}} I_{\{S(\mathbf{X})\geq\gamma\}} \ln f(\mathbf{X};\mathbf{v}). \tag{5}$$

This parameter can be estimated as:

$$\mathbf{v}^* = arg\max_{v} \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\mathbf{X}_i)\geq\gamma\}} \ln f(\mathbf{X}_i;\mathbf{v}). \tag{6}$$

where $\mathbf{X_i}$ are generated from pdf $f(.;\mathbf{v})$.

Thus the CE algorithm for optimization can be stated as follows:

General CE algorithm

1: Define an initial value $\mathbf{v_0}$, set $t = 1$.
2: Generate a sample $\mathbf{X_1}, \ldots, \mathbf{X_N}$ from density $f(.; \mathbf{v}_{t-1})$.
3: Calculate γ_t as $\gamma_t = \max\{s : \mathbb{P}_{v_{t-1}}(S(\mathbf{X}) \geq s) \geq \rho\}$
4: Calculate $\mathbf{v_t}$ using equation (6).
5: If for some $t \geq d$, $\gamma_t = \gamma_{t-1} = \ldots = \gamma_{t-d}$, then stop;
6: Set t=t+1 and reiterate from step 2.

Several approximations can be done to simplify the general CE algorithm:

First, the value $\gamma_t = \max\{s : \mathbb{P}_{v_{t-1}}(S(\mathbf{X}) \geq s) \geq \rho\}$ can be estimated from the sample $\mathbf{X_1}, \ldots, \mathbf{X_N}$ as follows:

Calculate the performances $S(\mathbf{X}_i)$ for all i, and sort them out in increasing order: $S_{(1)} \leq S_{(2)} \leq \ldots S_{(N)}$, and then evaluate the $(1-\rho)$ sample quantile as

$$\hat{\gamma}_t = S_{(\lceil (1-\rho)N \rceil)} \tag{7}$$

A further simplification in the general CE algorithm can be obtained for 0-1 combinatorial optimization problems, if we consider the solution vector $\mathbf{X} = (X_1, \ldots, X_n)$ to be Bernoulli random variables, $\mathbf{X} \sim Ber(\mathbf{p})$. In this case the pdf is

$$f(\mathbf{X}; \mathbf{p}) = \prod_{i=1}^{n} p_i^{X_i} \left(1 - p_i\right)^{1-X_i}, \tag{8}$$

since each X_j can only be 0 or 1,

$$\frac{\partial}{\partial p_j} \ln f(\mathbf{X}; \mathbf{p}) = \frac{X_j}{p_j} - \frac{1 - X_j}{1 - p_j} = \frac{1}{(1-p_j)p_j}\left(X_j - p_j\right). \tag{9}$$

using this result, we can calculate the maximum in Equation (6) by setting the first derivative with respect to p_j to zero for all j:

$$\frac{\partial}{\partial p_j} \sum_{i=1}^{N} I_{\{S(\mathbf{X}_i \geq \gamma)\}} \ln f(\mathbf{X}_i; \mathbf{p}) = \frac{1}{(1-p_j)p_j} \sum_{i=1}^{N} I_{\{S(\mathbf{X}_i \geq \gamma)\}}(X_{ij} - p_j) = 0. \tag{10}$$

and finally we obtain the following expression for the estimation of p_j:

$$p_j = \frac{\sum_{i=1}^{N} I_{\{S(\mathbf{X}_i \geq \gamma)\}} X_{ij}}{\sum_{i=1}^{N} I_{\{S(\mathbf{X}_i \geq \gamma)\}}}. \tag{11}$$

The CE algorithm for optimization of 0-1 problems can be stated now as follows:

CE algorithm for combinatorial optimization

1: Define an initial value $\mathbf{p_0}$, say $\mathbf{p_0} = (\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2})$; set $t = 1$.
2: Generate a sample $\mathbf{X_1}, \ldots, \mathbf{X_N}$ of bernoulli vectors with success probability $\mathbf{p}_{t-1}$.

3: Calculate the performances $S(\mathbf{X}_i)$ for all i.
4: Sort the performances in increasing order $S_{(1)} \leq S_{(2)} \leq \ldots S_{(N)}$.
5: Estimate $\hat{\gamma}_t$ using Equation (7).
6: Estimate $\hat{\mathbf{p}}_t = (\hat{p}_{t,1}, \hat{p}_{t,2}, \ldots, \hat{p}_{t,n})$ using equation (11).
7: If for some $t \geq d$, $\gamma_t = \gamma_{t-1} = \ldots = \gamma_{t-d}$, then stop;
8: Set t=t+1 and reiterate from step 2.

Several improvements have been proposed to this algorithm, like the *smoothed updating* [7], consisting in estimating vector $\hat{\mathbf{p}}_t$ using Equation (11) followed by a smoothing step:

$$p_{t,j} = \alpha p_{t,j} + (1 - \alpha)\, p_{t-1,j} \tag{12}$$

3 The Hybrid CE-HNN Algorithm

As we describe in the section 2.1, the binary HNN repairs the unfeasible solutions of a combinatorial optimization problem with binary encoding, obtaining feasible solutions that fulfill the COP's constraints. However the HNN cannot ensure that these feasible solutions are optimum with regard to a given objective function. For this reason, many hybrid heuristics have been proposed using a HNN as local heuristic. As an example, Figure 1 shows the search space of a optimization problem where only a small portion are feasible solutions due to the inherent problem constraints. In this example, the HNN acts as a "filter" of unfeasible solutions, producing solutions in the correct region of the search space. Many algorithms have been used as global search heuristics to be mixed with HNNs, for example Genetic Algorithms [10]-[13] or Simulated Annealing [14], [15].

In our case, we present a new hybrid heuristic algorithm formed by the binary HNN described in Section 2.1 and the CE method described in the section 2.2, in order to solve optimization problems with binary encoding and constraints among the components of the solution. In this algorithm the solutions are encoded by means of a vector of bits with $0 - 1$ values, which we denominate x.

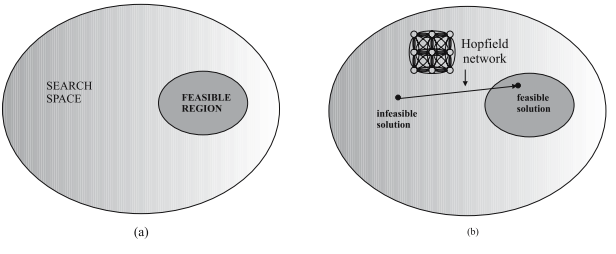


Fig. 1. Example of the effect of a Hopfield network in reducing the search space size: the unfeasible solutions are "filtered", producing feasible ones

Given a solution x, the CE method considers each component of the sequence as a random variable with Bernoulli pdf. Firstly, the CE initializes the vector x to $x = \{\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}\}$. Then, using the vector x as probabilities, the CE generates a sample of possible solutions. Note that usually these solutions are not feasible solutions. Thus, these solutions are repaired by means of the binary HNN, obtaining feasible solutions which replace the original ones. Following to this step, the objective function is evaluated for each feasible solution and the obtained values are sorted in ascending order (maximization problem). Then the parameter γ of the expression (7) is re-calculated. Finally, the CE also recalculates the probabilities of the vector x by means of the Equation (11) and applying la Equation (12). This procedure is recursive, and finalizes when the parameter γ has the same value k times in a row (k is a design parameter). Figure 2 shows an schema of the hybrid HNN-CE. Note that the Hopfield network repairs the solutions generated by means of the probabilities vector x, being the generated feasible ones used to update the search parameters of the CE method, i.e. γ and x.

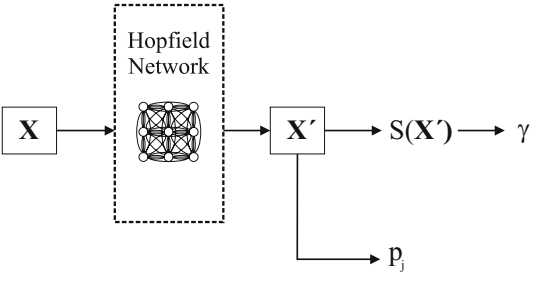


Fig. 2. Schema of the hybrid HNN-CE algorithm

4 Experiments in the Generalized Frequency Assignment Problem

The Generalized Frequency Assignment Problem (GFAP hereafter) in a mobile communications network can be stated as follows: given a mobile communication network formed by N cells, and a set of M available frequencies, achieve an assignment of frequencies to every cell, in such a way that the system is free from interferences, and the number of assigned frequencies is maximized.

The minimum distance between frequencies in the system in order to avoid interferences is given by an integer matrix D. The solution of the problem can be represented as an $N \times M$ binary matrix X, where the rows represents the cells and the columns represents the available frequencies. An element $x_{ij} = 1$ means that the frequency j has been assigned to the cell i. Some articles which have dealt with this problem before are [16], [18].

Mathematically the problem can be defined as follows:

Find an assignment X, which maximizes

$$\max\left(f(X)=\sum_{i=1}^{N}\sum_{j=1}^{M}x_{ij}\right) \tag{13}$$

and such as, if $x_{ik}=1$ and $x_{jl}=1$,

$$|k-l|\geq d_{ij} \tag{14}$$

Note that the definition of GFAP coincides with the type of problem which is possible to solve directly by means of the hybrid HNN-CE, i.e, binary encoding and constraints among the components of the solution. In this case, the vector x which represents the solution in the HNN-CE is formed by the elements of the matrix of frequencies X through the next equation:

$$x_{(i-1)\cdot m+j}=X_{ij},\qquad i=1,\ldots,n; j=1,\ldots,m \tag{15}$$

in such a way that exists a equivalence between the matrix and the vector and the Hopfield network can be represented as a matrix with the neurons x_{ij}. Furthermore, the dynamics of the Hopfield network depends on the matrix D which define the minimum distance between a neuron with state "1" and the next neuron in the same column with state "1" too. In consequence, the equation of input to the neuron and the updating function (see equation 2) depends on the matrix D according to the following equations:

$$u_{ik}(t)=isgn\left(\sum_{\substack{p=1\\p\neq i}}^{n}\sum_{\substack{q=max(1,k-D_{i,p}+1)\\q\neq k}}^{min(m,k+D_{i,p})}x_{pq}\right)\ \forall\ i,\ k, \tag{16}$$

$$\phi(u_{ik})=\begin{cases}0 & if\quad u_{ik}>0\\ 1 & otherwise.\end{cases} \tag{17}$$

In this way the HNN gives feasible solutions, i.e. solutions which fulfill the Equation (14), while the CE method searches for the optimal among these feasible solutions.

Four problems of different sizes have been studied, though the evolution of the algorithm has only been analyzed in instance 4 defined in [16]. We will compare our algorithm with other hybrid approach which uses a genetic algorithm in order to optimize the problem (HNN-GA). The Hopfield network is the same in the two algorithms so we can verify the performance of both algorithms in similar simulation conditions.

The parameters which configure the hybrid approaches are: in the CE algorithm we use the parameter $1-\rho$ (quantile) of 0.95, N (number of samples in

each iteration) of 100 and the stop parameter fixed to 5 [7]. In the genetic algorithm we use a population of 100 individuals for 200 generations, roulette wheel selection, 2-point cross and mutation by change of bit [17].

Table 1 shows the main characteristics of the problems GFAP which we use. The first problem instance considered is a small one, the second can be considered a medium one and others ones are large GFAP instances. The fourth instance (marked by a *) has been considered previously in several articles about frequency assignment [16], [18] and [19] and it will be used to analyze the convergence properties of the algorithms.

Table 1. Main characteristics of GPAG instances considered

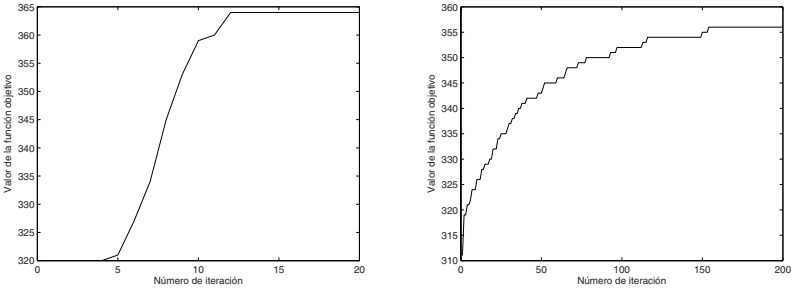
Problem	N	M
1	10	20
2	20	40
3	30	60
4*	25	73

Table 2 shows the results of 30 simulations of each problem by means of the average value and the best fitness obtained. It can be verified how in instances 1 and 2 both algorithms present similar behavior, but when the size of the problem is incremented (instances 3 and 4) the CE algorithm performs better than the HNN-GA. This shows that the CE algorithm presents better scalability than the HNN-GA.

Table 2. Average and best results for 30 experiments in each instances

Problem	HNN-EC (average)	HNN-EC (best)	HNN-GA (average)	HNN-GA (best)
1	77.4	**80**	78.0	**80**
2	194.1	**200**	197.4	**200**
3	357.6	**363**	338.6	349
4*	359.6	**364**	350.3	356

Other important point is the convergence of the algorithms. In order to analyze it we use the fourth instance. Figures 3 (a) and (b) shows an example of the convergence of both algorithms. First, note that, whereas in the HNN-GA the calculation of the average is very simple because the number of generations is constant in the 30 experiments, the HNN-CE algorithm is stopped when the parameter γ has the same value 5 times in a row. This is why we only show the experiment in which the best solution was obtained. We can see that the HNN-CE algorithm has a faster convergence than the HNN-GA. Note that the HNN-GA needed more than 150 generations in order to converge (100 individuals by generation) whereas the HNN-CE needed only 13 iterations (100 samples by iteration).



(a) (b)

Fig. 3. Example of convergence of HNN-CE and HNN-GA approaches for problem 4; (a) HNN-CE convergence; (b) HNN-GA convergence

In Figure 3 is possible to see how the two hybrid algorithms follow the typical dynamics of the global algorithms that form them. For example, the genetic algorithm has an evolution with small improvements each few generations, typical characteristic of this algorithm. However, the hybrid HNN-CE has a trajectory with a larger slope, i.e. good improvements in a short time. The dynamics of a HNN-CE rare time extends beyond 30 iterations, though this point depends on the values of the parameters used in the algorithm.

5 Conclusions

In this paper we have presented a hybrid algorithm based on the Cross-entropy method and a Hopfield neural network. The description of the hybrid algorithm and its main properties has been discussed. The performance of this approach has been shown by solving a combinatorial optimization problem known as the generalized assignment problem, where the HNN-CE algorithm outperforms another hybrid approach which includes a genetic algorithm.

References

1. Talbi, E.: A taxonomy of hybrid metaheuristics. Journal of Heuristics 8, 541–564 (2002)
2. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Computing Surveys 35, 268–308 (2003)
3. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: model, taxonomy, and design issues. IEEE Trans. Evol. Comput. 9(5), 474–488 (2005)
4. Rubinstein, R.Y.: The simulated entropy method for combinatorial and continous optimization. Methodology and computing in applied probability 2, 127–190 (2002)
5. Rubinstein, R.Y., Kroese, D.P.: The cross-entropy method: a unified approach to combinatorial optimization, Monte-carlo simulation and machine learning. Springer, NY (2004)

6. Hopfield, J.: Neurons and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. 79, 2554–2558 (1982)
7. De Boer, P., Kroese, D.P., Mannor, S., Rubinstein, R.Y.: A tutorial on the cross-entropy method. Annals of Operations Research 134, 19–67 (2005)
8. Shrivastava, Y., Dasgupta, S., Reddy, S.: Guaranteed convergence in a class of Hopfield networks. IEEE Trans. Neural Networks 3, 951–961 (1992)
9. Salcedo-Sanz, S., Bousoño-Calzón, C.: A portable and scalable algorithm for a class of constrained combinatorial optimization problems. Computers & Operations Research 32, 2671–2687 (2005)
10. Balicki, J., Stateczny, A., Zak, B.: Genetic algorithms and Hopfield neural networks to solve combinatorial optimization problems. Applied Mathematics and Computer Science 10(3), 568–592 (1997)
11. Watanabe, Y., Mizuguchi, N., Fujii, Y.: Solving optimization problems by using a Hopfield neural network and genetic algorithm combination. Systems and Computers in Japan 29(10), 68–73 (1998)
12. Salcedo-Sanz, S., Bousoño-Calzón, C.: A hybrid neural-genetic algorithm for the frequency assignment problem in satellite communications. Applied Intelligence 22, 207–218 (2005)
13. Bousoño-Calzón, C., Figueiras-Vidal, A.R.: Emergent techniques for dynamic frequency assignment: merging genetic algorithms and neural networks. In: Aerospace. Proc. of the RTO IST Symposium on Frequency Assignment, Sharing and Conservation in Systems, vol. 32, pp. 12/1–12/5 (1998)
14. Salcedo-Sanz, S., Santiago-Mozos, R., Bousoño-Calzón, C.: A hybrid Hopfield network-simulated annealing approach for frequency assignment in satellite communications systems. IEEE Trans. Systems, Man and Cybernetics, part B 34(2), 1108–1116 (2004)
15. Calderón-Macías, C., Sen, M.K., Stoffa, P.L.: Hopfield neural networks, and mean field annealing for seismic deconvolution and multiple attenuation. Geophysics 62(3), 992–1002 (1997)
16. Funabiki, N., Takefuji, Y.: A neural network parallel algorithm for channel assignment problems in cellular radio networks. IEEE Trans. Veh. Technol. 42(4), 430–437 (1992)
17. Goldberg, D.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading, MA (1989)
18. Lai, W.K., Coghill, G.C.: Channel assignment through evolutionary optimization. IEEE Trans. Veh. Technol. 55(1), 91–95 (1996)
19. José-Revuelta, L.M.S.: A new adaptive genetic algorithm for fixed channel assignment. Information Sciences 177(13), 2655–2678 (2007)

Author Index