Michael R. Berthold
John Shawe-Taylor
Nada Lavrač (Eds.)

Advances in Intelligent Data Analysis VII

LNCS 4723

7th International Symposium
on Intelligent Data Analysis, IDA 2007
Ljubljana, Slovenia, September 2007, Proceedings

IDA 07

Springer

Lecture Notes in Computer Science 4723

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

Editorial Board

Michael R. Berthold John Shawe-Taylor
Nada Lavrač (Eds.)

Advances in Intelligent Data Analysis VII

7th International Symposium
on Intelligent Data Analysis, IDA 2007
Ljubljana, Slovenia, September 6-8, 2007
Proceedings

Springer

Volume Editors

Michael R. Berthold
University of Konstanz, Department of Computer and Information Science
Box M 712, 78457 Konstanz, Germany
E-mail: michael.berthold@uni-konstanz.de

John Shawe-Taylor
University College London
Centre for Computational Statistics and Machine Learning
Department of Computer Science, Gower Street, London WC1E 6BT, UK
E-mail: jst@cs.ucl.ac.uk

Nada Lavrač
Jožef Stefan Institute, Department of Knowledge Technologies
Jamova 39, 1000 Ljubljana, Slovenia
E-mail: nada.lavrac@ijs.si

Library of Congress Control Number: 2007934038

CR Subject Classification (1998): H.3, I.2, G.3, I.5.1, I.4.5, J.2, J.1, J.3

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

ISSN 0302-9743
ISBN-10 3-540-74824-5 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-74824-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12120369 06/3180 5 4 3 2 1 0

Preface

We are proud to present the proceedings of the seventh biennial conference in the Intelligent Data Analysis series. The conference took place in Ljubljana, Slovenia, September 6-8, 2007. IDA continues to expand its scope, quality and size. It started as a small side-symposium as part of a larger conference in 1995 in Baden-Baden (Germany). It quickly attracted more interest in both submissions and attendance as it moved to London (1997) and then Amsterdam (1999). The next three meetings were held in Lisbon (2001), Berlin (2003) and then Madrid in 2005. The improving quality of the submissions has enabled the organizers to assemble programs of ever-increasing consistency and quality. This year we made a rigorous selection of 33 papers out of almost 100 submissions. The resulting oral presentations were then scheduled in a single-track, two-and-a-half-day conference program, summarized in the book that you have before you.

In accordance with the stated IDA goal of "bringing together researchers from diverse disciplines," we believe we have achieved an excellent balance of presentations from the more theoretical – both statistical and machine learning – to the more application-oriented areas that illustrate how these techniques can be used in practice. For example, the proceedings include papers with theoretical contributions dealing with statistical approaches to sequence alignment as well as papers addressing practical problems in the areas of text classification and medical data analysis. It is reassuring to see that IDA continues to bring such diverse areas together, thus helping to cross-fertilize these fields.

Organizing a conference such as IDA is not possible without the assistance and support of many individuals. We are particularly grateful to Tina Anžič and Heather Fyson, who worked tirelessly behind the scenes. Ingrid Fischer worked with Richard van de Stadt to make sure the proceedings were finished flawlessly in time for the meeting. But, crucially, putting together a program of assured quality was only possible through the detailed refereeing of the members of the Program Committee, many of whom also submitted papers and attended the conference.

September 2007

Michael R. Berthold
John Shawe-Taylor
Nada Lavrač

Conference Organization

General Chair	Nada Lavrač Jožef Stefan Institute Ljubljana, Slovenia
Program Chairs	Michael R. Berthold University of Konstanz Konstanz, Germany
	John Shawe-Taylor University College London London, UK
Conference Chair	Mitja Jermol Jožef Stefan Institute Ljubljana, Slovenia
Local Chair	Tina Anžič Jožef Stefan Institute Ljubljana, Slovenia
Registration Chair	Špela Sitar Jožef Stefan Institute Ljubljana, Slovenia
Publication Chair	Ingrid Fischer University of Konstanz Konstanz, Germany
Publicity Chairs	Marjana Plukavec Jožef Stefan Institute Ljubljana, Slovenia
	Alan Tucker Brunel University London, UK
Webmaster	Peter Burger University of Konstanz Konstanz, Germany
Tutorial Chair	Frank Klawonn Fachhochschule Braunschweig/Wolfenbüttel Wolfenbüttel, Germany

Student Grant and Awards Chair	Joost N. Kok Leiden University Leiden, The Netherlands
Finance Chair	Heather Fyson University of Konstanz Konstanz, Germany

Program Committee

Niall Adams, Imperial College London, UK
Martin Atzmüller, University of Würzburg, Germany
Bettina Berendt, Humboldt University Berlin, Germany
Daniel Berrar, University of Ulster, Northern Ireland, UK
Guillaume Beslon, INSA-Lyon, LIRIS, France
Prasad Bhanu, Florida A&M University, USA
Christian Borgelt, European Centre for Soft Computing, Spain
Jean-François Boulicaut, INSA-Lyon, LIRIS UMR CNRS, France
Pavel Brazdil, University of Porto, Portugal
Michael Böhlen, Free University of Bozen-Bolzano, Italy
Klemens Böhm, Universität Karlsruhe, Germany
Luis de Campos, Universidad de Granada, Spain
Bob Clark, Tripos, Inc., USA
Fabio Crestani, University of Lugano, Switzerland
Oscar Cubo-Medina, Laboratorio de SSOO (DATSI), Spain
Luc De Raedt, Katholieke Universiteit Leuven, Belgium
Giuseppe Di Fatta, The University of Reading, UK
Werner Dubitzky, University of Ulster, UK
Sašo Džeroski, Jožef Stefan Institute, Slovenia
André Elisseeff, IBM Research, Switzerland
Fazel Famili, National Research Council of Canada, Canada
Jason Farquhar, Max Planck Institute for Biological Cybernetics, Germany
Ad Feelders, Universiteit Utrecht, The Netherlands
Fridtjof Feldbusch, Universität Karlsruhe, Germany
Ingrid Fischer, University of Konstanz, Germany
Douglas Fisher, Vanderbilt University, USA
Alex Freitas, University of Kent, UK
Elisa Fromont, Katholieke Universiteit Leuven, Belgium
Elisabeth Gassiat, Université Paris-Sud 11, France
Fosca Giannotti, KDDLAB, ISTI-CNR, Pisa, Italy
Santiago González, Universidad Politécnica de Madrid, Spain
Marko Grobelnik, Jožef Stefan Institute, Ljubljana, Slovenia
Gabriela Guimaraes, CENTRIA UNL/FCT, Portugal
Larry Hall, University of South Florida, USA
Pilar Herrero, Universidad Politécnica de Madrid, Spain

Alexander Hinneburg, Martin Luther University Halle-Wittenberg, Germany
Frank Höppner, University of Applied Sciences BS/WF, Germany
Daniel Keim, University of Konstanz, Germany
Frank Klawonn, University of Applied Sciences BW/WF, Germany
Joost N. Kok, Leiden University, The Netherlands
Paul Krause, University of Surrey, UK
Rudolf Kruse, University of Magdeburg, Germany
Matti Kääriäinen, University of Helsinki, Finland
Antonio LaTorre, Universidad Politécnica de Madrid, Spain
Pedro Larrañaga, University of the Basque Country, Spain
Wee Sun Lee, National University of Singapore, Singapore
Hans-J. Lenz, Instute of Statistics and Econometrics, Germany
Xiaohui Liu, Brunel University, United Kingdom
Sofian Maabout, LaBRI Université Bordeaux 1, France
Trevor Martin, University of Bristol, United Kingdom
Dunja Mladenic, Jožef Stefan Institute, Ljubljana, Slovenia
Igor Mozetič, Jožef Stefan Institute, Ljubljana, Slovenia
Susana Nascimento, Universidade Nova de Lisboa, Portugal
Olfa Nasraoui, University of Louisville, USA
Detlef Nauck, BT Intelligent Systems Research Centre, United Kingdom
Andreas Nürnberger, University of Magdeburg, Germany
Tim Oates, CSEE University of Maryland, USA
Francesco d'Ovidio, University of Bari, Italy
Simon Parsons, City University of New York, USA
Kristiaan Pelckmans, Katholieke Universiteit Leuven, Belgium
José-Maria Peña, Technical University of Madrid, Spain
Tuan Pham, James Cook University, Australia
Marco Ramoni, Harvard Medical School, United States
Celine Robardet, LIRIS/INSA-Lyon, France
Víctor Robles Forcada, Universidad Politécnica de Madrid, Spain
Roberta Siciliano, University of Naples, Italy
Arno Siebes, Universiteit Utrecht, Netherlands
Rosaria Silipo, Spoken Translation Inc., Germany
Myra Spiliopoulou, Otto-von-Guericke-University, Magdeburg, Germany
Martin Spott, BTexact Technologies, United Kingdom
Fay Sudweeks, Murdoch University, Australia
Luis Talavera, Universitat Politécnica de Catalunya, Spain
Ljupčo Todorovski, Jožef Stefan Institute, Ljubljana, Slovenia
Hannu Toivonen, University of Helsinki, Finland
Antony Unwin, Augsburg University, Germany
Reinhard Viertl, Vienna University of Technology, Austria
Stefan Wrobel, Fraunhofer IAIS & University of Bonn, Germany
Hong Yan, City University of Hong Kong, Hong Kong, PR China
Daniel Yeung, IEEE SMC Research Institute, Hong Kong
Mohammed Zaki, Rensselaer Polytechnic Institute, USA

Referees

Maurizio Atzori
Robert Banfield
Mario Boley
Janez Brank
Bjoern Bringmann
Patrick Pak-Kei Chan
Giuseppe Delvecchio
Frank Eichinger
Patrik Hoyer
Aneta Ivanovska
Sandra Lima
Laura Madejón López
Wing W. Y. Ng
Siegfried Nijssen
Daniela Oelke
Bostjan Pajntar
Pance Panov
Aleksandar Peckov
Jean-Philippe Pellet
Friedrich Pukelsheim
Joern Schneidewind
Petteri Sevon
Larry Shoemaker
Ivica Slavkov
Matthias Steinbrecher
Dennis Wegener
Hartmut Ziegler

Table of Contents

Ranking

Trees

Sequence/ Time Series Analysis

Knowledge Discovery

Visualization

Text Mining

Bioinformatics

Applications

Compact and Understandable Descriptions of Mixtures of Bernoulli Distributions

Jaakko Hollmén and Jarkko Tikka

Helsinki Institute of Information Technology – HIIT
Helsinki University of Technology, Laboratory of Computer and Information Science, P.O. Box 5400, FI-02015 TKK, Espoo, Finland
Jaakko.Hollmen@tkk.fi, tikka@mail.cis.hut.fi

Abstract. Finite mixture models can be used in estimating complex, unknown probability distributions and also in clustering data. The parameters of the models form a complex representation and are not suitable for interpretation purposes as such. In this paper, we present a methodology to describe the finite mixture of multivariate Bernoulli distributions with a compact and understandable description. First, we cluster the data with the mixture model and subsequently extract the maximal frequent itemsets from the cluster-specific data sets. The mixture model is used to model the data set globally and the frequent itemsets model the marginal distributions of the partitioned data locally. We present the results in understandable terms that reflect the domain properties of the data. In our application of analyzing DNA copy number amplifications, the descriptions of amplification patterns are represented in nomenclature used in literature to report amplification patterns and generally used by domain experts in biology and medicine.

1 Introduction

In data analysis, the model should absorb the essentials about the data measured from a phenomenon and abstract away the irrelevant details about a particular data set. Parsimonious representations aim at particularly compact and simplified models. These kind of models offer an appealing basis for understanding and describing a phenomenon of interest. Previously, we have investigated parsimonious model representations in ecology [12], where we predicted nutrient concentrations in coniferous trees with a sparse regression methodology. In a time series prediction context, we have proposed a fast input selection method for long-term prediction [14] using a filter strategy. This strategy selects a possibly non-contiguous set of autoregressive variables with linear techniques and builds more complex non-linear prediction models using only the selected variables. In our experience, the parsimonious models are highly desired by domain experts, for instance, in biology, medicine, and ecology. In the models mentioned above, we have included roughly ten percent of the variables (in fact, parameters) compared to the models represented by all the parameters (full models). The sparse models still produce as accurate predictions as the full models. Another line of

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 1–12, 2007.

research where an attempt is made to concisely describe a data set is reported in [15]. We have presented a tool for automatically generating data survey reports for the modeler to be aware of the properties of the data set. While technically slightly different, the spirit still remains the same as in the current work: the focus is on describing the cluster structure and the contents of the clusters. The aim of the current paper is to present a way to summarize a finite mixture model for 0-1 data concisely and with a simple, domain-compatible representation.

Our research has been motivated by work in analyzing DNA copy number amplifications represented as 0-1 data by profiling [9] and by mixture modeling [13]. The mixture modeling approach offers an elegant way to model DNA amplification patterns in a probabilistic framework. However, the mixture models are summarized by arrays of numerical probability values that are hard to grasp. Therefore, we investigate how to describe the essential properties of the mixture models through the parameters of the models, or alternatively through the clustered data sets. Our proposed solution is based on the maximal frequent itemsets which are extracted from the clustered data sets. The descriptions are represented in the style of the descriptions used in literature to report amplification patterns and generally used by domain experts.

The rest of the paper is organized as follows: Sect. 2 describes the DNA copy number amplification data and our previous research in this context. Sect. 3 describes mixture models in the analysis of 0-1 data and the partitioning scheme for dividing the data in to cluster-specific data sets. The main topic of the paper — how to describe the mixture model for 0-1 data in a compact and understandable fashion — is explained in Sect. 4. Experiments are reported in Sect. 5 and the paper is summarized in Sect. 6. The nomenclature for the chromosome regions, which is used in the experimental part of the paper, is described in Appendix A.

2 DNA Copy Number Amplification Database

We have analyzed the database of DNA copy number amplifications collected with a bibliomics survey from 838 journal articles covering a publication period of ten years from 1992 until 2002 (for details, see [9]). DNA copy number amplifications are localized chromosomal aberrations that increase the number of copies of a chromosomal region from two to at least five. In the database, the DNA copy number amplifications are recorded for $N = 4590$ cancer patients in $d = 393$ chromosomal regions covering the whole human genome, and the observed data are the presence ($x_{ij} = 1$) or the absence ($x_{ij} = 0$) of DNA copy number amplifications for the patient i in the chromosomal region j, where $i = 1, \ldots, 4590$ and $j = 1, \ldots, 393$. For the case including only chromosome 1 presented later in the paper, the dimensions of the data are $N = 446$ and $d = 28$. The nomenclature for the chromosome regions used later in this paper is briefly described in Appendix A. In our previous work, we have analyzed a large 0-1 database of DNA copy number amplification patterns in human neoplasms [9]. We characterized the genome-wide data with cancer-specific amplification profiles with a probabilistic interpretation and

clustered the data with hierarchical clustering. Amplification-based clustering demonstrated that cancers with similar etiology, cell-of-origin or topographical location have a tendency to obtain convergent amplification profiles [9]. Furthermore, we applied independent component analysis (ICA) [7] to identify amplification hot spots, which are sparse, genome-wide factors defining statistically independent amplification sites in the data.

3 Mixture Models of DNA Copy Number Amplifications

Finite mixture models are widely used in data analysis and pattern recognition due to their flexibility and solid theoretical basis. The finite mixture model is parameterized by the number of components J and their corresponding mixing proportions π_j with non-negativity constraints $\pi_j \geq 0$ and necessary constraint $\sum_{j=1}^{J} \pi_j = 1$ so that the distribution integrates to one. Each of the component distributions is parameterized by a vector of parameters $\theta_j = (\theta_{j1}, \ldots, \theta_{jd})$. Considering the class of mixture models with J Bernoulli distribution components, each with dimensionality d, the model is summarized with $J + J \times d = J \times (1 + d)$ parameters and the appropriate conditional independence and independence assumptions. The probability of the data vector $\mathbf{x} = (x_1, \ldots, x_d)$ can be calculated as

$$P(\mathbf{x}) = \sum_{j=1}^{J} \pi_j \, P(\mathbf{x} \mid \theta_j) = \sum_{j=1}^{J} \pi_j \prod_{i=1}^{d} \theta_{ji}^{x_i} (1 - \theta_{ji})^{1 - x_i}. \quad (1)$$

Learning from the data is most conventionally done in the framework of maximum likelihood estimation using the iterative Expectation-Maximization (EM) algorithm [3,16].

Recently, we modeled a refined version of the DNA copy number amplification database containing only malignant tumors, or cancers, and modeled the DNA copy number amplification patterns with finite mixtures of Bernoulli distributions in a chromosome-specific manner [13]. Model selection was performed for each chromosome in order to select an appropriate number of component distributions using a 5-fold cross-validation procedure repeated ten times. As a result, we got 23 mixture models for chromosomes $1, 2, \ldots, 22, X$ with altogether 111 component distributions. The Y chromosome was left out of the analysis due to lack of data. In our example model for the DNA copy number amplifications in chromosome 1, we have 6 component distributions and the data dimension $d = 28$, so we have $6 \times (1 + 28) = 174$ parameters. The resulting model (see Fig. 1) is determined in terms of the maximum likelihood estimates of the parameters, which might be hard to interpret since the parameters of the model may have different roles and do not obey any simple geometric similarity observable by the human eye. The subsequent problem is to consider the result set and think of the best possible way to convey the model to experts in biology and medicine. The details of the mixture modeling is reported in [13]. Here, we concentrate on presenting the models with a compact and understandable description. The description also has a direct relevance to diagnostic patterns of amplification that can be used by experts in their research or clinical work.

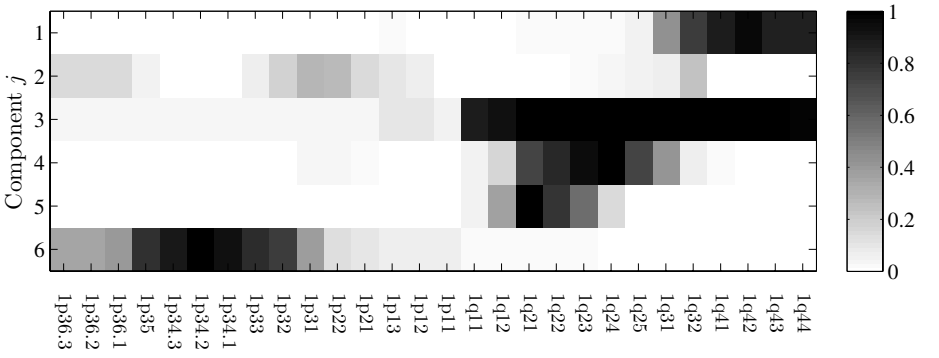


Fig. 1. Parameters $\theta_{ji}, j = 1, \ldots, 6,\ i = 1, \ldots, 28$ of the final mixture model. Each row represent the parameters of the corresponding component. The mixture proportions are $\pi_1 = 0.07, \pi_2 = 0.24, \pi_3 = 0.21, \pi_4 = 0.20, \pi_5 = 0.19$, and $\pi_6 = 0.09$. The names of the bands of the chromosome 1 (corresponding to 28 variables) are shown under the x-axis. For a brief explanation of the naming scheme of chromosomal regions, see Appendix A.

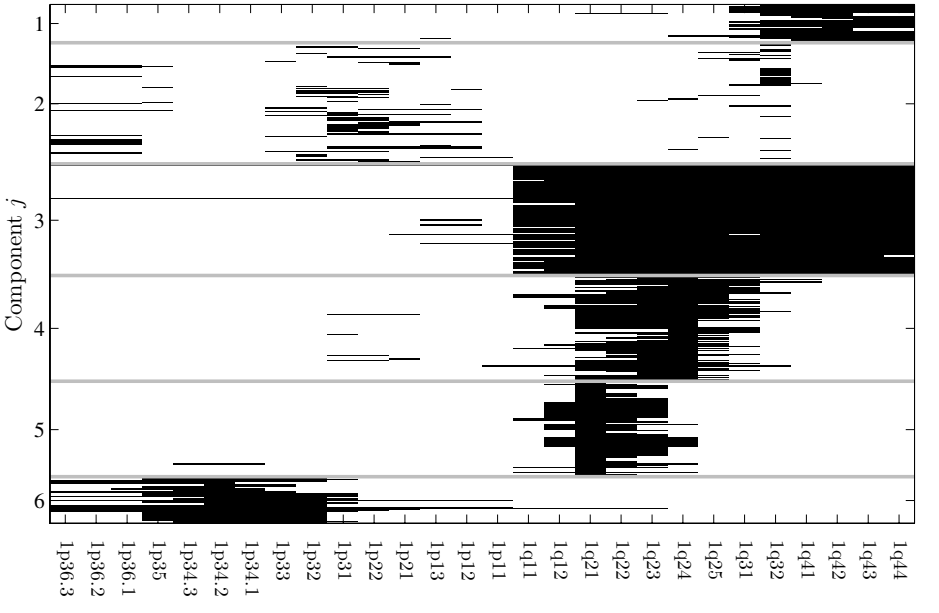


Fig. 2. The clustered DNA copy number amplification data is illustrated. Each horizontal line is one data vector; black areas mark ones and the white areas are zeroes in the data. The grey horizontal lines mark the partitions determined by the six probabilistic amplification patterns in the mixture model (see Fig. 1). The data in the clusters can easily be characterized, underlying the goodness of the clustering model and enabling compact and understandable cluster-specific descriptions of the model and the data.

The finite mixture model can be used in clustering by associating a cluster with each of the component distributions. Following the probabilistic approach, we can partition the original data set by allocating each data vector in to a cluster with the maximum posterior probability given the data vector. This maximum posterior allocation can be written with the help of Bayes's theorem as

$$P(j \mid \mathbf{x}) = \frac{P(j)P(\mathbf{x} \mid j)}{P(\mathbf{x})} = \frac{P(j)P(\mathbf{x} \mid j)}{\sum_{j'=1}^{J} P(j')P(\mathbf{x} \mid j')}. \tag{2}$$

Since the denominator in Eq. 2 is constant for all component distributions j, it is sufficient to partition the data vector $\mathbf{x}$ in to the cluster j^* using

$$j^* = \underset{j}{\operatorname{argmax}}\ P(j)P(\mathbf{x} \mid j) = \underset{j}{\operatorname{argmax}}\ \pi_j \prod_{i=1}^{d} \theta_{ji}^{x_i}(1-\theta_{ji})^{1-x_i}. \tag{3}$$

The clustering of the data set in to $J = 6$ partitions is illustrated in Fig. 2. In the following, we will consider the analysis of the clustered data.

4 Compact Description of the Mixture Models

We consider two alternative approaches to describe the multivariate Bernoulli distributions for modeling 0-1 data. The first approach is based on describing the model directly through the parameters of the mixture model (Sect. 4.1 and Sect. 4.2). The second approach to describe the model is with the help of the cluster-specific data sets that are achieved when the original data have been partitioned (Sect. 4.3). Our methodological contributions concentrate on the latter (describing the model though the data), but in the following, we also explore ways to characterize the model in terms of the parameters. It is also instructive to see how the methods are related to each other.

4.1 Modes of the Component Distributions

The simplest of all descriptions (and also the most compact) is to describe the model in terms of the modes of the respective component distributions. By definition, this only highlights the most probable elements of the data vectors described by the component distributions of the mixture model. If local chromosomal areas are sought for diagnostic purposes, this may be an ideal choice. However, this might suffer from the extremely simplified representation that is not faithful to the original data. For instance, if there are two distinct large probabilities, only one will be represented by the mode. For instance, component 2 of our example model in Fig. 1 can hardly be summarized by a mode, and neither can component 3, which has a large contiguous range of high probabilities.

4.2 Hypothetical Mean Organism

In bacteriology (see [5] and references), biochemical profiles of bacterial strain have been summarized by means of replacing probabilities θ_{ij} with their quantized counterparts a_{ij} as

$$a_{ij} = \begin{cases} 1 & \text{if } 1/2 \leq \theta_{ij} \leq 1, \\ 0 & \text{if } 0 \quad \leq \theta_{ij} < 1/2. \end{cases}$$

The profiles formed by the a_{ij} are called hypothetical mean organisms and abbreviated HMO. They represent the vectors of probabilities with their most likely realizations by quantizing each of the parameters to the most likely value of the each component of θ_j (either one or zero). One noteworthy aspect is that this could in fact be a non-existent case in the data set and more importantly, could be an impossible real-world situation. Nonetheless, it describes one realization that characterizes the parameters with the minimum quantization error.

4.3 Maximal Frequent Itemsets from Clustered Data

A conceptually different approach to describe the model is through the data sets that are created by partitioning the data by the maximum posterior rule in the hard clustering way (see Eq. 3). Then, any description of the cluster (and therefore the component distribution of that cluster), is calculated as a function of the clustered data sets. As mentioned before, the goal of the description is to be compact and understandable, so that domain experts can take full advantage of the modeling effort. Since our data is 0-1 data, any description must be specific to the nature of data. Frequent itemsets are one such description. Frequent itemsets are disjunctions of a set of attributes that co-occur in the data set.

As a typical example, imagine a market basket with possible items $X = \{\texttt{milk}, \texttt{butter}, \texttt{bread}\}$. If, for instance `milk` and `butter`, occur together in baskets with frequency of over 70 percent, we say that the $\{\texttt{milk}, \texttt{butter}\}$ itemset is 0.7-frequent. The Apriori algorithm for calculating the frequent itemsets is one of the classical algorithms [1,8] in data mining. However, if a market basket with items $\{\texttt{milk}, \texttt{butter}, \texttt{bread}\}$ is a frequent itemset with our pre-specified frequency threshold, all of its subsets are also frequent. This is the so-called anti-monotonicity property of frequent itemsets. For description purposes, this is somewhat wasteful, and certainly not compact. Instead, we turn our attention to maximal frequent itemsets [2], which are defined to be the largest (in cardinality) frequent itemsets that exceed the threshold. A set of maximal frequent itemsets is several orders of magnitude smaller than the usual set of frequent itemsets, making them ideal in providing compact descriptions of the items in the database [2]. In the above case, only the itemset $\{\texttt{milk}, \texttt{butter}, \texttt{bread}\}$ would be reported, and not any of its subsets. We have used an algorithm for finding maximal frequent itemsets called MAFIA [2]. The algorithm uses a search strategy that integrates a depth-first traversal of the itemset lattice with effective pruning mechanisms and it is especially efficient when the itemsets in the database are very long [2]. The need for mining maximal frequent itemsets becomes clear

when we look at the results, which are long indeed, and impractical to mine with the Apriori algorithm [1,8]. In our experiments, we have chosen to use the frequency threshold $\sigma = 0.5$, since then the extracted (maximal) frequent itemset would be present in the majority of cases in the cluster data. This could be further motivated by some majority voting protocol.

Previously, we have used the mixture of Bernoulli distributions in clustering 0-1 data in order to derive frequent itemsets from the cluster-specific data sets [6]. We investigated whether the frequent itemsets extracted in the clusters would be any different from the ones extracted globally. First, we clustered the data with a mixture of multivariate Bernoulli distributions and subsequently extracted frequent itemsets from the cluster-specific data sets. We introduced an L_1-norm based distance measure between sets of frequent itemsets and concluded that the frequent itemsets are markedly different from those extracted from the data set globally. On the basis of this previous work, we can expect different frequent itemsets to emerge from local, cluster-specific data sets and for them to describe the contents of the cluster more accurately. Frequent itemsets approximate in essence the marginal distribution of data, but can be also used as a basis in estimation of the joint distribution with the principle of maximum entropy [10,6].

The novelty in this paper compared to our previous work [6] is the real application that we are tackling, and we are trying to extract a compact and understandable description of the phenomenon. Furthermore, we are extracting descriptions of one model selected through a model selection procedure and not quantifying the differences over a range of models. Technically, we revert to maximal frequent itemsets and the translated descriptions based on those.

5 Experiments

We now describe the experimental work in training the mixture models from data and extracting maximal frequent itemsets from the cluster-specific data sets. The compact and understandable descriptions are based on the maximal frequent itemsets. The descriptions are presented in the original naming scheme for chromosomal regions. This nomenclature [11] is presented briefly in Appendix A. In our example chromosome 1, the chromosomal regions are coded with 28 regions based on the banding structure. The list of regions is: `1p36.3`, `1p36.2`, `1p36.1`, `1p35`, `1p34.3`, `1p34.2`, `1p34.1`, `1p33`, `1p32`, `1p31`, `1p22`, `1p21`, `1p13`, `1p12`, `1p11`, `1q11`, `1q12`, `1q21`, `1q22`, `1q23`, `1q24`, `1q25`, `1q31`, `1q32`, `1q41`, `1q42`, `1q43`, `1q44`. If we have an amplification of the three first chromosomal regions `1p36.3`, `1p36.2`, and `1p36.1` (denoted as a range `1p36.1`–`1p36.3`), the corresponding data vector would be $\mathbf{x} = (1, 1, 1, 0, \ldots, 0)$.

During the analysis of all chromosomes, the model selection procedure was used to select an appropriate complexity of the mixture model by varying the number of component distributions from $J = 2, \ldots, 20$. We trained 50 models by repeating 5-fold cross-validation ten times and selecting the model complexity with the largest validation log-likelihood. In one chromosome, this was seen to result in a model in which several component distributions were modeling

Table 1. Comparison of the different descriptions of the mixture of multivariate Bernoulli distributions for the chromosome 1 example. The parameters of the model are depicted in Fig. 1 and the clustered data in Fig. 2. The hypothetical mean organisms (HMO) are expressed in terms of the quantized probabilities a_{ij}. The modes of the component distributions are only given in cases when there is a clear single peak in the component distribution. Our compact descriptions (in the last column) are summarized as ranges from the contiguous maximal frequent itemsets extracted from the clustered data sets. The HMOs and our compact descriptions happen to describe the same chromosomal regions, although they are created using different paths.

j	HMO	mode	description based on itemsets
1	00000000000000000000000011111	1q42	1q32–1q44
2	00000000000000000000000000000	not given	none
3	00000000000000001111111111111	not given	1q11–1q44
4	00000000000000000011111000000	1q23	1q21–1q25
5	00000000000000000011100000000	1q21	1q21–1q23
6	00011111100000000000000000000	1p34.3	1p35–1p32

essentially the same data. In this case, a simpler model was selected by hand based on a hump of the validation likelihood. In all other chromosomes, a model selection procedure was followed.

5.1 Characterization of the Patterns

Repeating our proposed procedure for all chromosomes (except chromosome Y, which was discarded due to a low number of DNA copy number amplifications), we get in total 140 amplification patterns described by the maximal frequent itemsets with the frequency threshold $\sigma = 0.5$. The amplification patterns span a total of 337 chromosomal bands with some overlap between them. It is possible to have many itemsets for one component distribution; typically, they would be overlapping. The descriptions for the chromosome 1 example are listed in Table 1. To summarize the length distribution of all the itemsets, we extracted the following statistics: the average number of items in the itemsets is 5.1 and the median is 3. The 10th and 90th percentiles of the number of items are 1 and 12, respectively. The maximum number of items is 27. The large size of the individual itemsets underlines the impossibility of extracting frequent itemsets from the cluster-specific data: the number of all frequent itemsets (all subsets of the maximal frequent itemsets) would be about 146 million and would not be of any use in describing the models.

The mixture models seem to offer a nice way to model the domain with a probabilistic approach and with an excellent generalization ability, which is the ability to describe the general behavior of the amplification patterns. One characteristic of the identified models is their contiguous, smooth nature, which makes it very plausible for them to occur in nature. A surprising finding about the descriptions is that most of them describe contiguous chromosomal regions, that is,

spatially connected areas of the chromosome, although the extraction happens independently for the individual items. As the DNA copy number amplifications are expected to occur in the spatial manner [9], this speaks for the good quality of both the mixture model and the extracted descriptions.

As a reference, we extracted maximal frequent itemsets with a frequency threshold $\sigma = 0.5$ from all the data from chromosome 1 and got the following two itemsets: {1q21, 1q22} and {1q22, 1q23}. Another comparison would be to extract maximal frequent itemsets with $\sigma = 0.5/6$, the number 6 being the optimal number of clusters found with the aid of the model selection procedure. The resulting collection of itemsets was {1p31}, {1p22}, {1p36.1, 1p36.2, 1p36.3}, three overlapping itemsets with three items between 1p35 and 1p32, and a long itemset covering the whole 1q-arm. Two findings are striking: some spurious results emerge (1p36.1–1p36.3) and some results (1q-arm) shadow other interesting results found through partitioning of the data.

Another way to investigate the nature of the patterns is to compare them to some external data. One such data set that has close connections to DNA copy number amplifications is the fragile sites, which are discussed in more detail in our previous work [9]. There are 117 listed fragile sites in the genome (excluding the Y chromosome), of which 104 fragile sites map to our defined amplification patterns and 65 map to the ends of the amplification patterns. We would be interested to know if there is an unexpectedly large number of fragile sites in the ends of our amplification patterns, since this would indicate a possible explanation for DNA breakage associated with the amplification. We have compared the frequency of fragile sites in the ends of the amplification patterns and compared it with the frequency of fragile sites inside the patterns. A hypothesis test was executed with the help of a permutation test [4]. The frequency of fragile sites in the end regions is 0.3693 and inside the patterns it is 0.3086. Running the permutation test with 10000 repetitions, essentially sampling from the distribution of the null hypothesis by randomly picking 104 sites from the set of all possible sites, mapping them to the amplification patterns and calculating the frequency of border patterns and inside patterns, we get the differences for the random placements for the fragile sites. The p-value is calculated as a tail integral of the empirical distribution where the frequency of samples exceeds the true difference between the frequencies (one-tailed test). The resulting p-value is 0.0069, implying statistical significance of the findings. In absolute terms, the difference between the frequencies may not seem that large, but the relative difference is substantial. Thus, we claim scientifically relevant findings, as well.

6 Summary and Conclusions

Finite mixture models are a probabilistically sound and elegant way to model data, but can be hard to understand and describe compactly. In this paper, we have presented a way to describe the component distributions of the mixture models by describing the underlying cluster-specific data in terms of maximal frequent itemsets. The mixture model is used to model the whole data set as

a sum distribution in the global fashion and the frequent itemsets model the marginal distributions of the partitioned data in a local fashion; partitions coincide with the underlying structure of the data. In our case study in the analysis of DNA copy number amplification data, the cluster structure is well identified and the extracted maximal frequent itemsets summarize the marginal distributions in the clusters compactly. The descriptions are presented using the terminology used in literature, providing a compact and understandable summary of the models.

Acknowledgments

This work has been supported by the Academy of Finland in the Research Program SYSBIO (Systems Biology and Bioinformatics), grant number 207469. We thank Jouni Seppänen for insightful discussions on the methodological topics and Samuel Myllykangas for active and fruitful collaboration on cancer genomics. We also thank Paul Grouchy for comments on the manuscript.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the Twentieth International Conference on Very Large Data Bases (VLDB'94), pp. 487–499 (1994)
2. Burdick, D., Calimlim, M., Gehrke, J.: MAFIA: A maximal frequent itemset algorithm for transactional databases. In: Proceedings of the 17th International Conference on Data Engineering (ICDE'2001), pp. 443–452 (2001)
3. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm (with discussion). Journal of the Royal Statistical Society, Series B 39, 1–38 (1977)
4. Good, P.: Permutation Tests — A Practical Guide to Resampling Methods for Testing Hypotheses, 2nd edn. Springer, Heidelberg (2000)
5. Gyllenberg, M., Koski, T.: Probabilistic models for bacterial taxonomy. TUCS Technical Report No 325, Turku Centre of Computer Science (January 2000)
6. Hollmén, J., Seppänen, J.K., Mannila, H.: Mixture models and frequent sets: combining global and local methods for 0-1 data. In: Barbará, D., Kamath, C. (eds.) Proceedings of the Third SIAM International Conference on Data Mining, pp. 289–293. Society of Industrial and Applied Mathematics (2003)
7. Hyvärinen, A., Karhunen, J., Oja, E.: Independent component analysis. John Wiley & Sons, Chichester (2001)
8. Mannila, H., Toivonen, H., Verkamo, A.I.: Efficient algorithms for discovering association rules. In: Knowledge Discovery in databases: Papers form the AAAI-94 Workshop (KDD'94), pp. 181–192. AAAI Press (1994)
9. Myllykangas, S., Himberg, J., Böhling, T., Nagy, B., Hollmén, J., Knuutila, S.: DNA copy number amplification profiling of human neoplasms. Oncogene 25(55), 7324–7332 (2006)
10. Pavlov, D., Mannila, H., Smyth, P.: Beyond independence: Probabilistic models for query approximation on binary transaction data. IEEE Transactions on Knowledge and Data Engineering 15(6), 1409–1421 (2003)

11. Schaffer, L.G., Tommerup, N. (eds.): An International System for Human Cytogenetic Nomenclature. S. Karger, Basel (2005)
12. Sulkava, M., Tikka, J., Hollmén, J.: Sparse regression for analyzing the development of foliar nutrient concentrations in coniferous trees. Ecological Modeling 191(1), 118–130 (2006)
13. Tikka, J., Hollmén, J., Myllykangas, S.: Mixture modeling of DNA copy number amplification patterns in cancer. In: Sandoval, F., Prieto, A., Cabestany, J., Graña, M. (eds.) IWANN 2007. LNCS, vol. 4507, pp. 972–979. Springer, Heidelberg (2007)
14. Tikka, J., Lendasse, A., Hollmén, J.: Analysis of fast input selection: Application in time series prediction. In: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4132, pp. 161–170. Springer, Heidelberg (2006)
15. Vesanto, J., Hollmén, J.: An automated report generation tool for the data understanding phase. In: Abraham, A., Koeppen, M. (eds.) Proceedings of the First International Workshop on Hybrid Intelligent Systems (HIS'01), pp. 611–625. Springer, Heidelberg (2002)
16. Wolfe, J.W.: Pattern clustering by multivariate mixture analysis. Multivariate Behavioral Research 5, 329–350 (1970)

A Nomenclature for the Chromosomal Regions

The naming scheme for chromosomal regions will be briefly presented in order to understand the proposed description scheme of mixture models for 0-1 data in our case of DNA copy number amplifications. In essence, the terminology is used by domain experts in literature when addressing chromosomal regions. Idiograms of G-banding patterns for normal human chromosomes at five different levels of resolution are presented in [11].

Using the resolution of the data used in this paper, the whole human genome is divided in to 393 chromosomal regions, each with its own systematic name. The example chromosome, chromosome 1, is divided in to 28 regions. The nomenclature follows an irregular, hierarchical naming scheme, where each region may (or may not) be divided in to smaller subregions. The whole chromosome is denoted by `1`, which consists of two chromosome arms, the shorter arm named `1p` and the longer one named `1q`. The next levels will add numbers subsequent to `1p` and `1q`, as for instance in `1q12`. The name `1q12` tells us that the region belongs to chromosome `1`, arm `q`, and region `12`. The `1q12` is not divided any further in our resolution. Another example is the regions `1q21.1`, `1q21.2`, `1q21.3`, which are names of regions on a finer scale and where the part after the decimal period denotes sub-regions of the chromosome region `1q21`. Which regions are and which are not divided to finer subparts is determined by the resolution of the naming scheme and the properties of the genome. Chromosomal regions are often expressed as continuous ranges as for instance `1q21`–`1q23`.

The whole list of regions in the chromosome 1 is as follows: `1p36.3`, `1p36.2`, `1p36.1`, `1p35`, `1p34.3`, `1p34.2`, `1p34.1`, `1p33`, `1p32`, `1p31`, `1p22`, `1p21`, `1p13`, `1p12`, `1p11`, `1q11`, `1q12`, `1q21`, `1q22`, `1q23`, `1q24`, `1q25`, `1q31`, `1q32`, `1q41`, `1q42`, `1q43`, `1q44`. The DNA copy number amplification data expressed in terms of

chromosomal regions can be transformed to 0-1 data according to the amplification status of the corresponding regions $\mathbf{x} = (x_1, x_2, x_3, \ldots, x_{27}, x_{28}) = (x_{\texttt{1p36.3}}, x_{\texttt{1p36.2}}, x_{\texttt{1p36.1}}, \ldots, x_{\texttt{1q43}}, x_{\texttt{1q44}})$. The 0-1 data is used in the modeling; the proposed methodology produces compact descriptions of the models in the terminology originally used by domain experts.

Multiplicative Updates for L_1–Regularized Linear and Logistic Regression

Fei Sha[1], Y. Albert Park[2], and Lawrence K. Saul[2]

[1] Computer Science Division, University of California
Berkeley, CA 94720-1776
[2] Department of Computer Science and Engineering
UC San Diego, La Jolla, CA 92093-0404
feisha@cs.berkeley.edu, {yapark,saul}@cs.ucsd.edu

Abstract. Multiplicative update rules have proven useful in many areas of machine learning. Simple to implement, guaranteed to converge, they account in part for the widespread popularity of algorithms such as nonnegative matrix factorization and Expectation-Maximization. In this paper, we show how to derive multiplicative updates for problems in L_1-regularized linear and logistic regression. For L_1–regularized linear regression, the updates are derived by reformulating the required optimization as a problem in nonnegative quadratic programming (NQP). The dual of this problem, itself an instance of NQP, can also be solved using multiplicative updates; moreover, the observed duality gap can be used to bound the error of intermediate solutions. For L_1–regularized logistic regression, we derive similar updates using an iteratively reweighted least squares approach. We present illustrative experimental results and describe efficient implementations for large-scale problems of interest (e.g., with tens of thousands of examples and over one million features).

1 Introduction

The search for sparse solutions appears as a theme in many seemingly unrelated areas of statistical learning. These areas include, for example, large margin classification by support vector machines (SVMs) [16], unsupervised learning by nonnegative matrix factorization (NMF) [8], and linear and logistic regression with L_1–norm regularization [3,6,12,15]. Between the first two of these areas, there recently emerged an unexpected connection. In particular, it was shown [13] that the main optimization in SVMs—an instance of nonnegative quadratic programming (NQP)—could be solved by a generalization of certain multiplicative updates proposed for NMF [8].

In this paper, we establish another connection in this framework. Specifically, we show that the same multiplicative updates developed for SVMs can also be used for L_1–regularized linear and logistic regression. The advantages of multiplicative updates for learning sparse representations [5,8] also transfer directly to this new setting.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 13–24, 2007.

The multiplicative updates that we study have two particularly appealing features. First, they are very simple to implement, with no tunable parameters or ad-hoc heuristics needed to ensure convergence. Second, they provide a guarantee of monotonic convergence, decreasing their loss functions at each iteration.

The transparency and reliability of these updates make them attractive for many applications in machine learning. In many real-world applications, it is necessary to modify or monitor the core optimizations; sometimes, they must be re-implemented on different platforms or distributed across multiple processors. These needs are not well-served by complicated black-box solvers.

The multiplicative updates we study in this paper have proven useful in many settings. In their simplest form, originally derived for NMF [8], the updates have been widely adopted for unsupervised learning and feature extraction. In their more general form, originally derived for SVMs [14], they have also been applied to problems in acoustic echo cancellation [10] and astrophysical data analysis [2]. We believe that these multiplicative updates will prove similarly attractive to many practitioners of L_1–regularized regression.

The paper is organized as follows. In section 2, we review the multiplicative updates proposed by [14] for nonnegative quadratic programming. We also show how to bound the error of intermediate solutions using ideas from convex duality. These types of guarantees have not been discussed in earlier work [13] on multiplicative updates. In sections 3 and 4, we describe how these updates are used for linear and logistic regression with L_1–norm regularization. These applications of multiplicative updates to L_1–norm regularized prediction also represent a novel contribution beyond previous work [13]. Finally, in section 5, we highlight several recent approaches most closely related to our own and conclude by discussing future directions for research.

2 Background in NQP

The problem of nonnegative quadratic programming (NQP) takes the form:

$$\begin{array}{ll} \text{Minimize} & f(\mathbf{v}) = \frac{1}{2}\mathbf{v}^\top \mathbf{A}\mathbf{v} + \mathbf{b}^\top \mathbf{v} \\ \text{subject to} & \mathbf{v} \geq \mathbf{0} \ . \end{array} \tag{1}$$

The notation $\mathbf{v} \geq \mathbf{0}$ is used to indicate that all the elements of $\mathbf{v}$ are required to be nonnegative. For simplicity, we assume that the matrix $\mathbf{A}$ in eq. (1) is strictly positive definite, and hence there exists a unique global minimum $\mathbf{v}^* \geq \mathbf{0}$ to this problem. Due to the nonnegativity constraints in eq. (1), its solution cannot be found in closed form. Thus, an iterative approach is required. Section 2.1 reviews the multiplicative updates proposed by Sha et al [13,14] for these problems in NQP. Section 2.2 shows that multiplicative updates can also be used to solve the Lagrange dual problems, yielding bounds on the error of intermediate solutions. To our knowledge, this aspect of convex duality has not been exploited in previous work on multiplicative updates.

2.1 Multiplicative Updates

The multiplicative updates for NQP are written in terms of matrices $\mathbf{A}^+$ and $\mathbf{A}^-$ that store the positive and negative elements of $\mathbf{A}$. In particular, we define:

$$A_{ij}^+ = \begin{cases} A_{ij} \text{ if } A_{ij} > 0, \\ 0 \quad \text{otherwise.} \end{cases} \quad A_{ij}^- = \begin{cases} |A_{ij}| \text{ if } A_{ij} < 0, \\ 0 \quad \text{otherwise.} \end{cases}$$

It follows from these definitions that $\mathbf{A} = \mathbf{A}^+ - \mathbf{A}^-$. The multiplicative updates for NQP involve matrix-vector products between $\mathbf{A}^+$ and $\mathbf{A}^-$ and the current estimate $\mathbf{v}$. As shorthand, we define vectors with elements $a_i = (\mathbf{A}^+\mathbf{v})_i$ and $c_i = (\mathbf{A}^-\mathbf{v})_i$. In terms of the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \Re^d$, the multiplicative updates take the simple closed form:

$$v_i \longleftarrow \left[\frac{-b_i + \sqrt{b_i^2 + 4a_i c_i}}{2a_i} \right] v_i \ . \tag{2}$$

The updates assume that the vector $\mathbf{v}$ is initialized with strictly positive elements. Then, as shown in [13], these updates converge monotonically to the global minimum of eq. (1), decreasing the loss function at each iteration.

Implementation of the updates is straightforward. In many applications, the updates can be performed by just a few lines of MATLAB code. Each update requires two matrix-vector multiplications for $\mathbf{A}^+\mathbf{v}$ and $\mathbf{A}^-\mathbf{v}$, essentially twice the computation required to evaluate the gradient. When the matrix $\mathbf{A}$ is itself nonnegative, eq. (2) reduces to previously derived updates for nonnegative matrix factorization [8], now widely used for unsupervised learning and feature extraction.

2.2 Convex Duality

By minimizing the Lagrangian associated with eq. (1), we obtain the Lagrange dual function [1]:

$$g(\boldsymbol{\lambda}) = \inf_{\mathbf{v} \in \Re^d} \left[f(\mathbf{v}) - \boldsymbol{\lambda}^\top \mathbf{v} \right]. \tag{3}$$

The Lagrange dual function can be used to obtain lower bounds on the solution of eq. (1). In particular, for any nonnegative pair $\mathbf{v}, \boldsymbol{\lambda} \geq \mathbf{0}$, we have:

$$g(\boldsymbol{\lambda}) = \inf_{\mathbf{u} \in \Re^d} \left[f(\mathbf{u}) - \boldsymbol{\lambda}^\top \mathbf{u} \right] \leq \left[f(\mathbf{v}) - \boldsymbol{\lambda}^\top \mathbf{v} \right] \leq f(\mathbf{v}). \tag{4}$$

The tightest lower bound in eq. (4) is obtained by maximizing the Lagrange dual function over $\boldsymbol{\lambda} \geq \mathbf{0}$. For the NQP in eq. (1), assuming the matrix $\mathbf{A}$ is invertible, the Lagrange dual problem is given by:

$$\begin{array}{ll} \text{Maximize} & g(\boldsymbol{\lambda}) = -\frac{1}{2} (\boldsymbol{\lambda} - \mathbf{b})^\top \mathbf{A}^{-1} (\boldsymbol{\lambda} - \mathbf{b}) \\ \text{subject to} & \boldsymbol{\lambda} \geq \mathbf{0} \ . \end{array} \tag{5}$$

Note that the Lagrange dual problem is also an instance of NQP, whose global maximum $\boldsymbol{\lambda}^* \geq \mathbf{0}$ can be computed using multiplicative updates. Finally, we note that all problems in NQP exhibit strong duality [1], guaranteeing that $g(\boldsymbol{\lambda}^*) = f(\mathbf{v}^*)$. By solving primal and dual problems in parallel, we can therefore bound the error of intermediate solutions that are obtained from multiplicative updates. We will use the observed duality gap $[f(\mathbf{v}) - g(\boldsymbol{\lambda})]$ for this purpose in section 3, when we reformulate L_1–regularized linear regression as an instance of NQP. For example, Fig. 1 shows the observed duality gap as a function of the number of multiplicative updates.

3 L_1–Regularized Linear Regression

In this section, we show how to use the multiplicative updates in eq. (2) for the problem of linear regression with L_1–norm regularization [3,15]. The training data for linear regression consists of labeled examples $\{(\mathbf{x}_\alpha, y_\alpha)\}_{\alpha=1}^n$, where $\mathbf{x}_\alpha \in \mathbb{R}^d$ and $y_\alpha \in \Re$. The L_1–regularized loss is given by:

$$\text{LOSS}(\mathbf{w}) \;=\; \frac{1}{2n}\sum_{\alpha=1}^{n}\left(y_\alpha - \mathbf{w}^\top \mathbf{x}_\alpha\right)^2 \;+\; \gamma \sum_{i=1}^{d} |w_i|, \tag{6}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the weight vector and $\gamma \geq 0$ is the regularization parameter. Let $\mathbf{w}^*$ denote the weight vector that minimizes the L_1–regularized loss. The second term on the right hand side of eq. (6) encourages sparse solutions to this problem; in particular, larger values of γ lead to increasing numbers of zeros among the elements of $\mathbf{w}^*$.

To cast L_1–regularized linear regression as an instance of NQP, specifically in the form of eq. (1), we define: $\mathbf{A} = \frac{1}{n}\sum_{\alpha=1}^{n} \mathbf{x}_\alpha \mathbf{x}_\alpha{}^\top$ and $\mathbf{b} = -\frac{1}{n}\sum_{\alpha=1}^{n} y_\alpha \mathbf{x}_\alpha$. With these definitions, we can rewrite the L_1–regularized loss in eq. (6) up to an additive constant as:

$$\mathcal{L}(\mathbf{w}) \;=\; \frac{1}{2}\mathbf{w}^\top \mathbf{A}\mathbf{w} + \mathbf{b}^\top \mathbf{w} + \gamma \sum_{i=1}^{d} |w_i| \tag{7}$$

Section 3.1 describes how to minimize the right hand side of eq. (7) by solving a problem in NQP. Section 3.2 derives the special structure of the Lagrange dual problem for this NQP. Finally, section 3.3 presents several illustrative experimental results.

3.1 Primal Formulation as NQP

We reformulate the optimization of the L_1–regularized loss by decomposing $\mathbf{w}$ into its positive and negative components. In particular, we introduce nonnegative variables $\mathbf{u}$ and $\mathbf{v}$ such that:

$$\mathbf{w} = \mathbf{u} - \mathbf{v}, \quad \mathbf{u} \geq \mathbf{0}, \quad \mathbf{v} \geq \mathbf{0} \;. \tag{8}$$

As shorthand notation, we also let $\boldsymbol{\gamma} \in \Re^d$ denote the vector whose every element is equal to the scalar regularizer γ in eq. (7). Finally, in terms of the variables $\mathbf{u}$ and $\mathbf{v}$, we consider the optimization:

$$\begin{array}{ll} \text{Minimize} & f(\mathbf{u},\mathbf{v}) = \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top \mathbf{A}(\mathbf{u}-\mathbf{v}) + \mathbf{b}^\top(\mathbf{u}-\mathbf{v}) + \boldsymbol{\gamma}^\top(\mathbf{u}+\mathbf{v}) \\ \text{subject to} & \mathbf{u} \geq \mathbf{0},\ \mathbf{v} \geq \mathbf{0}\ . \end{array} \tag{9}$$

Let $\begin{bmatrix}\mathbf{u}^* \\ \mathbf{v}^*\end{bmatrix}$ denote the minimizing solution of eq. (9). It is straightforward to show that either $u_i^* = 0$ and/or $v_i^* = 0$ at this minimum, due to the effect of the regularizer $\boldsymbol{\gamma}^\top(\mathbf{u}+\mathbf{v})$. It follows that $u_i^* + v_i^* = |u_i^* - v_i^*|$, and hence the minimum of eq. (9) maps directly onto the minimum of eq. (7). Thus we can use one problem to solve the other.

The change of variables in eq. (8) follows the strategy suggested by Koh et al [6], transforming the non-differentiable objective function in eq. (7) to the differentiable one in eq. (9). Here, the change of variables casts the problem of L_1–regularized linear regression as an instance of NQP. The NQP problem in eq. (9) can be solved using the multiplicative updates from section 2. Note that in this context, the updates are applied to the $2d$-dimensional nonnegative vector $\begin{bmatrix}\mathbf{u} \\ \mathbf{v}\end{bmatrix}$ obtained by concatenating the elements of $\mathbf{u} \in \Re^d_+$ and $\mathbf{v} \in \Re^d_+$.

3.2 Dual Formulation as BQP

By minimizing the Lagrangian associated with eq. (9), we obtain the Lagrange dual function:

$$g(\boldsymbol{\theta},\boldsymbol{\lambda}) \ = \ \inf_{\mathbf{u},\mathbf{v}\in\Re^d} \Big[f(\mathbf{u},\mathbf{v}) - \boldsymbol{\theta}^\top\mathbf{u} - \boldsymbol{\lambda}^\top\mathbf{v} \Big] \tag{10}$$

In general, the right hand side of eq. (10) is unbounded below, as can be seen by evaluating it in the special case that $\mathbf{u} = \mathbf{v}$:

$$\Big[f(\mathbf{v},\mathbf{v}) - \boldsymbol{\theta}^\top\mathbf{v} - \boldsymbol{\lambda}^\top\mathbf{v} \Big] \ = \ (2\boldsymbol{\gamma} - \boldsymbol{\theta} - \boldsymbol{\lambda})^\top\mathbf{v} \tag{11}$$

Setting $\mathbf{u} = \mathbf{v}$, the right hand side of eq. (10) thus reduces to a single linear term in $\mathbf{v}$. A finite minimum does not exist because we can scale the magnitude of $\mathbf{v}$ in eq. (11) to be arbitrarily large. The Lagrange dual function $g(\boldsymbol{\theta},\boldsymbol{\lambda})$ does, however, have a well-defined minimum in the special case that $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ are chosen precisely to cancel the divergence in eq. (11). In particular, suppose that:

$$\boldsymbol{\lambda} + \boldsymbol{\theta} = 2\boldsymbol{\gamma}, \tag{12}$$

which causes the right hand side of eq. (11) to vanish. Enforcing this constraint, and substituting $f(\mathbf{u},\mathbf{v})$ from eq. (9) into eq. (10), we find that the variables $\mathbf{u}$ and $\mathbf{v}$ appear in the Lagrangian only through their difference $\mathbf{w} = \mathbf{u}-\mathbf{v}$. In particular, with these substitutions, the minimization in eq. (10) reduces to:

$$g(\boldsymbol{\theta},\boldsymbol{\lambda})|_{\boldsymbol{\lambda}+\boldsymbol{\theta}=2\boldsymbol{\gamma}} = \inf_{\mathbf{w}\in\Re^d} \left[\frac{1}{2}\mathbf{w}^\top\mathbf{A}\mathbf{w} \ + \ \frac{1}{2}\left(2\mathbf{b} + \boldsymbol{\lambda} - \boldsymbol{\theta}\right)^\top\mathbf{w} \right]. \tag{13}$$

The quadratic form in eq. (13) yields a simple minimization. Thus, collecting the two different regimes (bounded and unbounded) of the dual, we have:

$$g(\boldsymbol{\theta}, \boldsymbol{\lambda}) = \begin{cases} -\frac{1}{8}\,(2\mathbf{b} + \boldsymbol{\lambda} - \boldsymbol{\theta})^\top \mathbf{A}^{-1} (2\mathbf{b} + \boldsymbol{\lambda} - \boldsymbol{\theta}) & \text{if } \boldsymbol{\lambda} + \boldsymbol{\theta} = 2\boldsymbol{\gamma}, \\ -\infty & \text{otherwise.} \end{cases} \tag{14}$$

As we shall see, the existence of these different regimes gives rise to a Lagrange dual problem with more structure than generic instances of NQP.

We can maximize the Lagrange dual function $g(\boldsymbol{\theta}, \boldsymbol{\lambda})$ over all nonnegative $\boldsymbol{\theta}, \boldsymbol{\lambda} \geq \mathbf{0}$ to derive a lower bound on the primal optimization in eq. (9). Clearly, for this maximization we only need to consider the domain over which the function $g(\boldsymbol{\theta}, \boldsymbol{\lambda})$ is finite and bounded below. Over this domain, we can use the equality constraint in eq. (12) to eliminate the variable $\boldsymbol{\lambda}$ and obtain the simpler maximization:

$$\begin{aligned} &\text{Maximize } h(\boldsymbol{\theta}) = -\tfrac{1}{2}\,(\boldsymbol{\theta} - \mathbf{b} - \boldsymbol{\gamma})^\top \mathbf{A}^{-1} (\boldsymbol{\theta} - \mathbf{b} - \boldsymbol{\gamma}) \\ &\text{subject to } \mathbf{0} \leq \boldsymbol{\theta} \leq 2\boldsymbol{\gamma}. \end{aligned} \tag{15}$$

This optimization is an instance of box quadratic programming (BQP), since in addition to the nonnegativity constraint on $\boldsymbol{\theta}$, there also appears the box constraint implied by eq. (12). The optimization can be solved using a variant of the multiplicative updates reviewed in section 2. The updates for BQP include a clipping operation to enforce the upper bound $\boldsymbol{\theta} \leq 2\boldsymbol{\gamma}$; for further details, see [13,14].

Note how the special structure of the primal optimization in eq. (9) is manifested in its dual Lagrange problem, eq. (15). In general, as shown in section 2, primal optimizations in NQP have dual optimizations in NQP, with both optimizations over variables of the same dimensionality. Here, however, the NQP in eq. (9) over the joint variables $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d_+$ generates as its Lagrange dual problem a smaller instance of BQP over the single variable $\boldsymbol{\theta} \in \mathbb{R}^d_+$.

3.3 Experimental Results

We experimented with the multiplicative updates for NQP to investigate their performance on problems in L_1–regularized linear regression. For these experiments, we created artificial data sets $\{(\mathbf{x}_\alpha, y_\alpha)\}_{\alpha=1}^n$ with inputs of varying dimensionality $d \sim 10^{2-3}$.

Each data set was created as follows. First, we randomly generated a "ground truth" weight vector $\mathbf{w} \in \Re^d$ with precisely $d/3$ negative elements, $d/3$ zero elements, and $d/3$ positive elements. The magnitudes of nonzero elements in this weight vector were sampled uniformly from the unit interval. For ease of visualization, the elements of $\mathbf{w}$ were also sorted from smallest to largest. (See the top left panel of Fig. 2.) Second, we randomly generated n inputs by sampling the elements of each input vector $\mathbf{x}_\alpha$ from a zero-mean Gaussian with unit variance. Finally, we generated n outputs by sampling each y_α from a Gaussian distribution with mean $\mu_\alpha = \mathbf{w}^\top \mathbf{x}_\alpha$ and standard deviation 0.2σ, where σ measured the standard deviation of the means $\{\mu_\alpha\}_{\alpha=1}^n$.

The noise in these data sets prevents a linear regression from exactly recovering the ground truth weight vector. The use of L_1–norm regularization, however, encourages sparse solutions, so that an L_1–regularized linear regression may be expected to yield an estimated weight vector with the same (or nearly the same) sparsity pattern. In the experiments reported below, the data sets had $n = 2d$ examples (so as to scale with the dimensionality of the inputs), and we set the regularization parameter to $\gamma = 0.1$. In this regime, L_1–norm regularization had the desired effect of encouraging appropriately sparse solutions. Our experiments were designed to measure the convergence of the multiplicative updates in this regime.

First, we present typical results from L_1–regularized linear regression on data with input dimensionality $d = 48$. Fig. 1 shows the observed duality gap between the primal and dual optimizations in eqs. (9) and (15) as a function of the number of multiplicative updates. Similarly, Fig. 2 shows the convergence of the weight vector $\mathbf{w} = \mathbf{u} - \mathbf{v}$ obtained from the primal optimization. For this figure, the elements of the weight vector were initialized at random. Also shown in the plot are the "ground truth" weight vector used to generate this data set, as well as the weight vector obtained from a linear regression without L_1–norm regularization. In both figures, it can be seen that the multiplicative updates converge reliably to the global minimum.

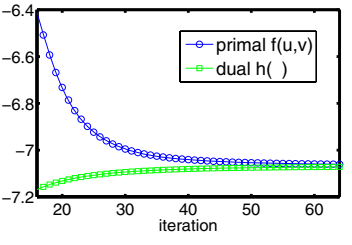


Fig. 1. Convergence of multiplicative updates for primal and dual optimizations in L_1–regularized linear regression; see text in section 3.3 for details

Next we present results on problems of varying input dimensionality d. We generated random data sets (as described above) with inputs of dimensionality $d = 48, 96, 192, 384, 768$, and 1536. We generated twelve data sets for each input dimensionality and averaged our results across these twelve data sets. For these

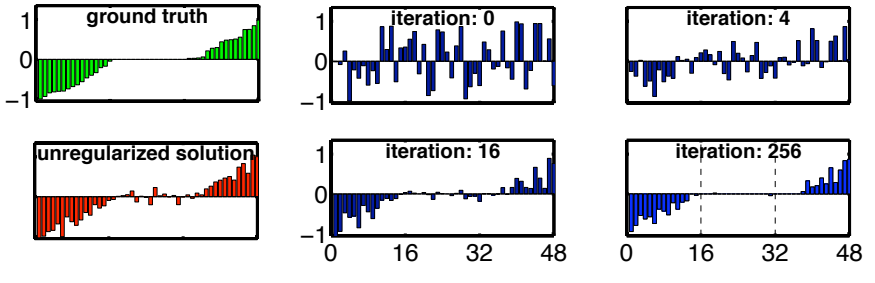


Fig. 2. Evolution of weight vector $\mathbf{w} \in \Re^{48}$ under multiplicative updates for L_1–regularized linear regression, starting from random initialization. Also shown are the "ground truth" weight vector (that would have been recovered in the absence of noise) and unregularized solution.

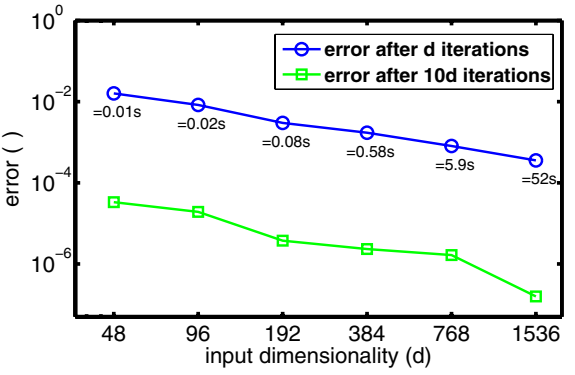


Fig. 3. Error ε_t from eq. (16) after $t = d$ and $t = 10d$ iterations of the multiplicative updates, on data sets of input dimensionality d. Each result represents an average over twelve different, randomly generated data sets. For the experiments with $t = d$ iterations, we also indicate the CPU times τ in seconds (per data set).

experiments, the weight vector was initialized by performing a linear regression without L_1–norm regularization: namely, $\mathbf{w}_0 = \mathbf{A}^{-1}\mathbf{b}$, with $\mathbf{A}$ and $\mathbf{b}$ defined as in eq. (7). We measured the convergence of the multiplicative updates as follows. Let $\mathbf{u}_t$, $\mathbf{v}_t$, and $\boldsymbol{\theta}_t$ denote the vectors obtained after t updates on the primal and dual optimizations in eqs. (9) and (15). Also, let $\mathbf{w}_t = \mathbf{u}_t - \mathbf{v}_t$. We computed the error ratio:

$$\varepsilon_t = \frac{\mathcal{L}(\mathbf{w}_t) - h(\boldsymbol{\theta}_t)}{\mathcal{L}(\mathbf{w}_0) - h(\boldsymbol{\theta}_t)}. \tag{16}$$

The numerator in eq. (16) simply measures the observed duality gap, while the ratio provides an easily computed upper bound on the amount of relative improvement $\frac{\mathcal{L}(\mathbf{w}_t) - \mathcal{L}(\mathbf{w}^*)}{\mathcal{L}(\mathbf{w}_0) - \mathcal{L}(\mathbf{w}^*)}$. Note that computing eq. (16) does not require exact knowledge of $\mathcal{L}(\mathbf{w}^*)$. We report the ratio in eq. (16), as opposed to the absolute value of the duality gap, because it normalizes to some extent for the degree of regularization and the corresponding difficulty of the optimization.

The results of these experiments are summarized in Fig. 3. On data sets of varying dimensionality d, the figure shows the error ε_t after different numbers of iterations t. The results in the figure were averaged over twelve randomly generated data sets. The figure show the average error ε_t after $t = d$ and $t = 10d$ iterations of the multiplicative updates: that is, after a number of iterations equal to and ten times greater than the input dimensionality. Again, it is seen that the multiplicative updates converge reliably and quickly to the global minimum. In terms of computation, each iteration of the updates involves four matrix-vector multiplications (two for the primal, two for the dual), but no matrix inversions or matrix-matrix multiplications. The figure also shows representative CPU times τ per data set, in MATLAB, on a Mac Pro workstation with a 2×3 GHz Dual-Core Intel Xeon processor. For t iterations of the updates, we expect $\tau = O(td^2)$, which is generally observed for medium to large values of d.

4 L_1–Regularized Logistic Regression

In this section, we show how to use multiplicative updates for L_1–regularized logistic regression. The training data consists of labeled examples $\{(\mathbf{x}_\alpha, y_\alpha)\}_{\alpha=1}^n$, where $\mathbf{x}_\alpha \in \mathbb{R}^d$ and $y_\alpha \in \{0, 1\}$. Let $s_\alpha = 2y_\alpha - 1$ denote the negatively and positively labeled examples by their signs $\{-1, +1\}$. The L_1–regularized log-loss is given by:

$$\mathcal{L}(\mathbf{w}) \;=\; -\frac{1}{n}\sum_{\alpha=1}^{n} \log\sigma\left(s_\alpha \mathbf{w}^\top \mathbf{x}_\alpha\right) \;+\; \gamma \sum_{i=1}^{d} |w_i|, \tag{17}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the weight vector, $\sigma(z) = [1 + e^{-z}]^{-1}$ is the logistic function, and $\gamma \geq 0$ is the regularization parameter.

Many algorithms for logistic regression are based on local quadratic approximations [9] to the log-loss in the neighborhood of the current estimate $\mathbf{w}$. These quadratic approximations generate iteratively reweighted least squares (IRLS) sub-problems that can be solved using simpler methods. The simplest quadratic approximation is obtained by expanding the first term on the right hand side in eq. (17) by its Taylor series. In our work, we use a different quadratic approximation that instead provides a provably global upper bound on the log-loss. Our approximation relies on an inequality originally introduced in the context of Bayesian logistic regression [4]:

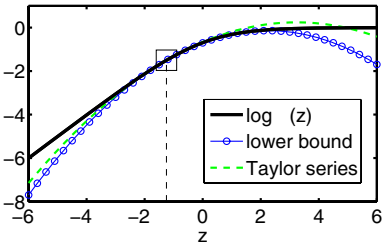


Fig. 4. Comparison of $\log\sigma(z)$, its quadratic Taylor series approximation, and the lower bound in eq. (18). The three curves meet inside the square box. Note that the Taylor series approximation does not provide a lower bound.

$$\log\sigma(z') \;\geq\; \log\sigma(z) \;+\; \frac{1}{2}(z' - z) \;-\; \frac{\tanh(z/2)}{4z}(z'^2 - z^2). \tag{18}$$

Eq. (18) holds for all real-valued pairs (z, z') and reduces to an equality for $z' = z$. Fig. 4 illustrates the bound around the point $z = -\frac{5}{4}$. This bound provides a more controlled approximation to the loss function than the one obtained from its Taylor series: in particular, looseness in the bound leads us only to *underestimate* the progress made in optimizing the true loss function. Applying the quadratic bound in eq. (18) to eq. (17) in the neighborhood of the current estimate $\mathbf{w}$, we generate a L_1–regularized least squares sub-problem that can be solved using the multiplicative updates from section 3. In fact, it is not necessary to iterate the multiplicative updates on these least squares sub-problems to convergence. Instead, we perform just one multiplicative update, then recompute the local quadratic approximation before performing the next one.

We experimented with the 20 NEWSGROUPS data set, currently available at the web site `http://people.csail.mit.edu/jrennie/20Newsgroups`. We attempted

Table 1. Percentage of zero elements in the weight vector and CPU times for L_1–regularized logistic regression on the NEWGROUP data set

regularizer (γ)	sparsity (%)	time (sec)
0.010	89.76	1296
0.025	92.24	1436
0.050	92.92	1218

regularizer (γ)	sparsity (%)	time (sec)
0.100	93.25	986
0.250	93.45	672
0.500	93.54	535

to replicate the set-up in [6]. Classifiers were trained to distinguish documents from newsgroups with names of the form `sci.*`, `comp.*`, and `misc.forsale` versus all the rest. We used the Bow toolkit [11] with the command "rainbow -g 3 -h -s -O 2 -i" to generate document feature vectors. This created 18,792 examples, each with 1,258,799 features. To manage the extremely high dimensionality of the feature space, we exploited the nonnegativity and sparseness of the feature vectors. In particular, by careful bookkeeping, it is possible to implement the multiplicative updates for L_1–regularized linear regression without having to construct or store the matrix $\mathbf{A}$ in eq. (7). Also, we only used multiplicative updates to solve the primal optimizations for L_1–regularized IRLS sub-problems: the dual optimizations were not constructed. The weight vectors were initialized from the results of unregularized logistic regressions, performed by limited-memory quasi-Newton methods (L-BFGS). For the L_1–regularized solutions, we terminated the multiplicative updates when the relative improvement of eq. (17) per iteration was less than 10^{-3} or when the training error dropped below 0.01%. Though not particularly stringent, these stopping criteria sufficed for the multiplicative updates to generate sparse solutions.

Table 1 shows results averaged over eight random 70/20/10 splits of the data into training, test, and development sets. The development set in each split was used to tune the value of the regularization parameter, γ. As expected, increasing values γ led to solutions of increasing sparsity. These solutions were found in

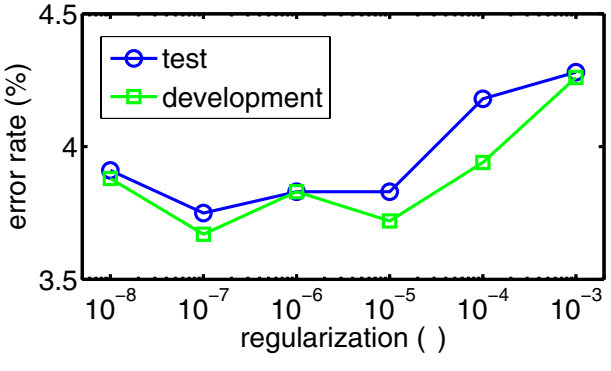


Fig. 5. Development and test error rates on one particular random split of the NEWSGROUP data. A value $\gamma = 10^{-7}$ is found that improves generalization in this split.

approximately 10-20 minutes of CPU time, demonstrating the feasibility of our approach for large-scale problems.

In half of the train/test/development splits of the NEWSGROUP data set, we observed that regularization led to improved generalization. For a typical one of these splits, Fig. 5 shows the error rates on the test and development sets as a function of the regularization parameter γ. It is possible that more consistent improvement across splits would have been observed using a finer search for the regularization parameter γ.

5 Discussion

There is a large literature on algorithms for L_1–regularized linear and logistic regression. Here, we highlight several recent approaches related to our own, as well as indicating important differences.

Closest in spirit to our approach are other bound optimization algorithms [7,10] which derive updates from an auxiliary function. In contrast to our approach, however, these other algorithms have non-multiplicative updates that involve matrix inverses. These matrix inverses are needed to re-estimate all of the elements of the weight vector $\mathbf{w} \in \Re^d$ in parallel (as opposed to simply performing coordinate descent). For large d, however, these matrix inverses may be prohibitively expensive.

Lee et al [9] pursue an iterative strategy for logistic regression that, like ours, is also based on quadratic approximations to the differentiable part of the log-loss. They use the LARS algorithm [3] to solve L_1-regularized least squares problems generated by second-order Taylor expansions. Our approach differs in using a variational bound for the quadratic approximation (see Fig. 4) and calling the multiplicative updates to solve the L_1–regularized least squares problems.

Koh et al [6] describe a fast, state-of-the-art interior point method [17] for L_1–regularized logistic regression. Our approach was directly inspired by two aspects of their work: first, the way they recast the L_1–norm regularizer as part of a differentiable objective function, and second, their impressive results on large-scale problems. (The NEWSGROUP data set analyzed in [6] is far larger than any of the data sets analyzed in [7,9,10].) Our approach differs from Koh et al [6] in that it provides a reasonably effective yet simpler re-estimation procedure. The multiplicative updates appear easier to implement, though not as fast to converge.

There are several important directions for future work. Though we have demonstrated the feasibility of our approach on a very large problem in logistic regression, with over one million features, further benchmarking is clearly needed. Also, we hope to study settings in which the updates are not merely used to solve isolated problems in L_1–regularized regression, but are embedded in larger models with more expressive power.

Acknowledgments

We are grateful to J. Blitzer (U. of Pennsylvania) for helping us to process the NEWSGROUP data set. This work was supported by NSF Award 0238323.

References

1. Boyd, S.P., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
2. Diego, J.M., Tegmark, M., Protopapas, P., Sandvik, H.B.: Combined reconstruction of weak and strong lensing data with WSLAP (2007), doi:10.1111/j.1365-2966.2007.11380.x
3. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. Annals of Statistics 32(2), 407–499 (2004)
4. Jaakkola, T., Jordan, M.I.: Bayesian parameter estimation via variational methods. Statistics and Computing 10, 25–37 (2000)
5. Kivinen, J., Warmuth, M.: Exponentiated gradient versus gradient descent for linear predictors. Information and Computation 132(1), 1–63 (1997)
6. Koh, K., Kim, S.-J., Boyd, S.P.: An interior-point method for large-scale ℓ_1–regularized logistic regression. JMLR 8, 1519–1555 (2007)
7. Krishnapuram, B., Carin, L., Figueiredo, M., Hartemink, A.: Sparse multinomial logistic regression: fast algorithms and generalization bounds. IEEE Transactions on Pattern Analysis and Intelligence 27(6), 957–968 (2005)
8. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) Advances in Neural Information Processing Systems, vol. 13, pp. 556–562. MIT Press, Cambridge, MA (2001)
9. Lee, S., Lee, H., Abbeel, P., Ng, A.Y.: Efficient l_1 regularized logistic regression. In: Proceedings of the Twenty First National Conference on Artificial Intelligence, Boston, MA (2006)
10. Lin, Y., Lee, D.D.: Bayesian l_1-norm sparse learning. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-06), Toulouse, France, vol. V, pp. 605–608 (2006)
11. McCallum, A.K.: Bow: a toolkit for statistical language modeling, text retrieval, classification and clustering (1996), http://www.cs.cmu.edu/~mccallum/bow
12. Ng, A.Y.: Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance. In: Proceedings of the Twenty First International Conference on Machine Learning (ICML-04), Banff, Canada, pp. 78–85 (2004)
13. Sha, F., Lin, Y., Saul, L.K., Lee, D.D.: Multiplicative updates for nonnegative quadratic programming. Neural Computation 19, 2004–2031 (2007)
14. Sha, F., Saul, L.K., Lee, D.D.: Multiplicative updates for large margin classifiers. In: Proceedings of the Sixteenth Annual Conference on Computational Learning Theory (COLT-03), Washington D.C., pp. 188–202 (2003)
15. Tibshirani, R.: Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society B 58(1), 267–288 (1996)
16. Vapnik, V.: Statistical Learning Theory. Wiley, N.Y. (1998)
17. Wright, S.J.: Primal-Dual Interior Point Methods. SIAM, Philadelphia, PA (1997)

Learning to Align: A Statistical Approach

Elisa Ricci[1], Tijl de Bie[2], and Nello Cristianini[2,3]

[1] Dept. of Electronic and Information Engineering, University of Perugia, 06125, Perugia, Italy
elisa.ricci@diei.unipg.it
[2] Dept. of Engineering Mathematics, University of Bristol, Bristol, BS8 1TR, UK
tijl.debie@gmail.com
[3] Dept. of Computer Science, University of Bristol, Bristol, BS8 1TR, UK
nello@support-vector.net

Abstract. We present a new machine learning approach to the inverse parametric sequence alignment problem: given as training examples a set of correct pairwise global alignments, find the parameter values that make these alignments optimal. We consider the *distribution* of the scores of all incorrect alignments, then we search for those parameters for which the score of the given alignments is as far as possible from this mean, measured in number of standard deviations. This normalized distance is called the 'Z-score' in statistics. We show that the Z-score is a function of the parameters and can be computed with efficient dynamic programs similar to the Needleman-Wunsch algorithm. We also show that maximizing the Z-score boils down to a simple quadratic program. Experimental results demonstrate the effectiveness of the proposed approach.

1 Introduction

We consider the problem of learning to align biological sequences given a training set of correct global alignments. Learning to align means learning the alignment parameters (the scoring matrix and gap costs) in such a way that the correct alignment has the best score among all possible alignments between two given sequences. This task is known as the *Inverse Parametric Sequence Alignment Problem* (IPSAP) introduced by Gusfield [4] and falls in the category of inverse parametric optimization problems [2]. Since its introduction, the problem has received considerable attention, e.g. in [5,6,7,9,10]. A similar task is also considered in [3] where a learning approach is used to determine the parameters of an unusual score function.

The most straightforward approach to IPSAP would be to consider a constraint for each incorrect alignment, and to explicitly specify that its score should be smaller than the score for the correct one. Such constraints can be shown to be linear in the alignment parameters, such that a linear programming method would suffice to find a feasible point satisfying all these constraints (assuming such a feasible point exists). However, a direct implementation of this strategy would be totally infeasible, as the number of incorrect alignments (and hence the number of constraints) is exponential in the length of the sequences.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 25–36, 2007.

A few recent publications [6], [7] have attacked this problem by adding constraints incrementally. Roughly their strategy can be summarized as follows. First an initial guess for the parameters is made. Then, the best alignment for this parameter setting is computed (using dynamic programming (DP)), and if it is not the optimal one, it is used in a constraint that specifies that its score should be worse than for the given, correct alignment. However, for each constraint to be added in this way an alignment needs to be computed, which quickly becomes expensive for long sequences and large numbers of iterations.

In this work, we approach the problem from a radically different perspective. Instead of considering incorrect alignments one by one as they turn out to have a better score for the current guess of the parameters, we consider *the full distribution* of the score of all possible alignments between the given sequence pairs. In particular, we compute the mean and the variance of this distribution as a function of the parameters. Importantly it can done exactly and efficiently, since a DP approach similar to the Needleman-Wunsch (NW) algorithm [8] is used. Then, we maximize the Z-score of the correct alignment subject to the alignment parameters, where the Z-score is defined as the number of standard deviations the score is away from its mean. In this way, the score of the correct alignment is separated as well as possible from the bulk of all alignment scores without considering each of these separately.

2 The Z-Score

Consider two strings $S_1 = s_1(1)\dots s_1(n_1)$ and $S_2 = s_2(1)\dots s_2(n_2)$ of lengths n_1 and n_2 respectively. The strings are ordered sequences of symbols $s \in \mathcal{S}$, with $\mathcal{S}$ a finite alphabet of size $n_{\mathcal{S}}$. In case of biological applications, the alphabet may contain the symbols associated to nucleotides ($\mathcal{S} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$) or to amino acids. For ease of notation, we will use the first $n_{\mathcal{S}}$ integers as the alphabet $\mathcal{S} = \{1, 2, \dots n_{\mathcal{S}}\}$.

Definition 1. *A global alignment $\mathcal{A}$ of two strings S_1 and S_2 is defined as a pair of strings $S_1' = s_1'(1)s_1'(2)\dots s_1'(n)$ and $S_2' = s_2'(1)s_2'(2)\dots s_2'(n)$ of length n for which $s_1'(i), s_2'(i) \in \mathcal{S}' = \mathcal{S} \cup \{-\}$, that are obtained by inserting the $-$ symbol anywhere in respectively S_1 and S_2, in such a way that there exists no i for which $s_1'(i) = s_2'(i) = -$.*

Every symbol $s_1'(i)$ corresponds to a specific symbol $s_2'(i)$. If these symbols are equal, we call this a *match*. If they are not equal, this is a *mismatch*. If one of the symbols is a $-$, this is called an *indel* or a *gap*. With each possible match, substitution or gap at position i, a score is attached. To quantify these scores, three score parameters can be used, one corresponding to a match (α_m), one to a mismatch (α_s), and one to a gap (α_g). The *score of the global alignment* can be expressed as a linear function of these parameters:

$$\phi\left(S_1, S_2, \mathcal{A}\right) = \alpha_m m + \alpha_s s + \alpha_g g = \boldsymbol{\alpha}^T \mathbf{x}$$

where we have defined the vectors $\boldsymbol{\alpha}^T = [\alpha_m\ \alpha_s\ \alpha_g]$ and $\mathbf{x}^T = [m\ s\ g]$ and m, s and g represent the number of matches, mismatches and gaps in the alignment.

In this model the penalty of a gap is fixed independently of the other gaps in the alignment. However for biological reasons it is often preferable to consider an affine function for gap penalties, i.e. to assign different costs if the gap starts (gap opening penalty α_o) in a given position or if it continues (gap extension penalty α_e). The score is also a linear function of the parameters, if $\boldsymbol{\alpha}^T = [\alpha_m\ \alpha_s\ \alpha_o\ \alpha_e]$ and $\mathbf{x}^T = [m\ s\ o\ e]$, with o and e being respectively the number of gap openings and gap extensions.

Since not all matches and mismatches are created equal, most often a symmetric *scoring matrix* $\boldsymbol{A} \in \Re^{n_S \times n_S}$ will be considered, which specifies different score values for matches and mismatches. For $1 \leq j, k \leq n_S$, $\boldsymbol{A}(j,k) = \boldsymbol{A}(k,j) = \alpha_{jk}$ represents the score associated with the correspondence between $s'_1(i) = j$ and $s'_2(i) = k$ or viceversa (thus, for $j = k$, there is a match, for $k \neq j$ there is a mismatch). In general we have $\frac{n_S(n_S+1)}{2}$ different parameters associated with the symbols of the alphabet plus two additional one corresponding to the gap penalties. We denote with q_{jk} the number of pairs with $s'_1(i) = j$ and $s'_2(i) = k$ and viceversa. Then the score is still $\phi(S_1, S_2, \mathcal{A}) = \boldsymbol{\alpha}^T \mathbf{x}$, with $\boldsymbol{\alpha}^T = [\alpha_{11}\ \alpha_{12}\ \ldots \alpha_{n_S n_S}\ \alpha_e\ \alpha_o]$ and $\mathbf{x}^T = [q_{11}\ q_{12}\ \ldots q_{n_S n_S}\ e\ o]$.

The number N of possible alignments $\mathcal{A}_j$ between S_1 and S_2 is clearly exponential in the sizes of the two strings. However, given a set of scoring parameters $\boldsymbol{\alpha}$, by the NW algorithm [8] the alignment with maximal score $\bar{\mathcal{A}}$ can be computed in $O(n_1 n_2)$-time. The optimal score is $\phi\left(S_1, S_2, \bar{\mathcal{A}}\right) = \boldsymbol{\alpha}^T \bar{\mathbf{x}}$, where for example in the three parameters model $\bar{\mathbf{x}}^T = [\bar{m}\ \bar{s}\ \bar{g}]$, contains the number of matches, mismatches and gaps in $\bar{\mathcal{A}}$.

Since the scoring function $\phi(S_1, S_2, \mathcal{A})$ is linear in the parameters also the mean $\mu(S_1, S_2)$ of the scores for all the alignments between S_1 and S_2 is linear:

$$\mu(S_1, S_2) = \frac{1}{N} \sum_{j=1}^{N} \phi(S_1, S_2, \mathcal{A}_j) = \boldsymbol{\alpha}^T \frac{1}{N} \sum_{j=1}^{N} \mathbf{x}_j = \boldsymbol{\alpha}^T \boldsymbol{\mu}_x$$

where $\boldsymbol{\mu}_x$ is the vector of the average values of $\mathbf{x}$ components by components. For example, in the three parameter model, $\boldsymbol{\mu}_x = [\mu_m\ \mu_s\ \mu_g]$ is the vector containing the average number of matches, mismatches and gap respectively. In a similar way, for the variance $\sigma^2(S_1, S_2)$ we have:

$$\sigma^2(S_1, S_2) = \frac{1}{N} \sum_{j=1}^{N} (\boldsymbol{\alpha}^T \mathbf{x}_j - \boldsymbol{\alpha}^T \boldsymbol{\mu}_x)^2 = \boldsymbol{\alpha}^T \boldsymbol{C} \boldsymbol{\alpha}$$

where $\boldsymbol{C}$ is a covariance matrix with elements:

$$c_{tk} = \frac{1}{N} \sum_{j=1}^{N} t_j k_j - \mu_t \mu_k = v_{tk} - \mu_t \mu_k \tag{1}$$

with $t, k \in \{m, s, g\}$ or $t, k \in \{q_{11}\ q_{12}\ \ldots q_{n_S n_S}\ o\ e\}$ respectively in the three parameters model and for an alphabet with n_S symbols and affine gap penalties.

Definition 2. *Let $\mu(S_1, S_2)$ and $\sigma^2(S_1, S_2)$ be the average score and the variance of the scores for all possible alignments between S_1 and S_2. We define the Z-score:*

$$Z(S_1, S_2) = \frac{\phi\left(S_1, S_2, \bar{\mathcal{A}}\right) - \mu(S_1, S_2)}{\sigma(S_1, S_2)} = \frac{\boldsymbol{\alpha}^T \boldsymbol{b}}{\sqrt{\boldsymbol{\alpha}^T \boldsymbol{C} \boldsymbol{\alpha}}} \tag{2}$$

with $\boldsymbol{b} = \bar{\boldsymbol{x}} - \boldsymbol{\mu}_x$

***Z*-score for a set of aligned sequence pairs.** More in general, we will consider a set of m pairs of sequences S_{1i} and S_{2i}, $i = 1 \ldots m$ along with an alignment $\bar{\mathcal{A}}_i$. In such case we can extend the alignment score by defining it as the sum of the alignments scores for each of the alignments in the set. The mean of this is the sum of the means for all sequence pairs (S_{1i}, S_{2i}) separately, and can be summarized by $\boldsymbol{b}^* = \sum_i \boldsymbol{b}_i$. Similarly, the variance is the sum of the variances: $\boldsymbol{C}^* = \sum_i \boldsymbol{C}_i$. Hence, the Z-score can be extended naturally to the case where there is more than one given aligned sequence pair by using for $\boldsymbol{b}^*$ and $\boldsymbol{C}^*$ instead of $\boldsymbol{b}$ and $\boldsymbol{C}$ in (2) above.

3 Fast and Exact Computation of the *Z*-Score

Given a pair of sequences S_1 and S_2, the computation of $\boldsymbol{b}$ and $\boldsymbol{C}$ is not trivial since the number of possible alignments N is exponential in the length of the sequences. In the following we show how DP can be used to obtain these values in a fast and efficient way.

The Simplest Scoring Scheme: Match, Mismatch, Gap. Assuming the previous notation, we have $\boldsymbol{b} = \bar{\mathbf{x}} - \boldsymbol{\mu}_x$. While $\bar{\mathbf{x}}$ is given, the components of $\boldsymbol{\mu}_x$ can be calculated exactly through DP. First, a matrix p is filled. Every cell $p(i, j)$ contains the number of all possible alignments between two prefixes of the strings S_1 and S_2. In fact each alignment corresponds to a path in the DP matrix. At the same time the DP tables for μ_m, μ_s and μ_g are gradually filled according to appropriate recursive relations. For example each element $\mu_m(i, j)$ is computed dividing the total number of matches by the number of alignments $p(i, j)$. If a match occur in position (i, j) $(M = 1)$ the total number of matches in (i, j) is obtained adding to the number of matches in the previous steps $(\mu_m(i, j-1)p(i, j-1)$, $\mu_m(i-1, j-1)p(i-1, j-1)$ and $\mu_m(i-1, j)p(i-1, j))$ $p(i-1, j-1)$ times a match. Once the algorithm is terminated, the mean values can be read in the cells $\mu_m(n_1, n_2)$, $\mu_s(n_1, n_2)$ and $\mu_g(n_1, n_2)$. In the same way also the elements of the covariance matrix $\boldsymbol{C}$ can be computed. This matrix is symmetric ($c_{sg} = c_{gs}$, $c_{mg} = c_{gm}$, $c_{sm} = c_{ms}$) and its elements can be obtained considering (1) with $t, k \in \{m, s, g\}$. The values of v_{tk} can be calculated, as for the mean values, with opportune recursive relations. Algorithm 1 shows in detail the computation of μ_m, while Algorithm 3 report the recursive relations for the other parameters (the associated initial conditions are not indicated due to lack of space).

Affine Gap. In a four parameter model (affine gap penalty), $\boldsymbol{C}$ is a 4x4 symmetric matrix with elements c_{tk} given by (1), with $t, k \in \{m, s, o, e\}$. The terms μ_m, μ_s, v_{mm}, v_{ms} and v_{ss} are calculated as above, while the other values are obtained with the formulas in Algorithm 3 in Appendix. The terms v_{se} and v_{so} are missing since they can be calculated with the same formulas of v_{me} and v_{mo} changing M with $1-M$ and μ_m with μ_s. Note that in some cases for low values of (i, j) some terms are not defined (i.e. $p(i, j-3)$ when $j = 2$). In such situations they must be ignored.

Due to lack of space an exhaustive explanation of recursive relations in Algorithm 3 is omitted. In order to just give the main idea behind them we provide an example. Suppose we want to compute the mean of the number of gap openings μ_o. Algorithm 1 with appropriate initial conditions ($\mu_o(0,0) = 0$, $\mu_o(i,0) = 1,\ \forall i = 1 \dots n_1$, $\mu_o(0,j) = 1,\ \forall j = 1 \dots n_2$) is used. Each element of the DP table $\mu_o(i,j)$ is computed dividing the total number of gap openings by the number of alignments $p(i,j)$. The total number of gap openings in (i,j) is obtained adding to the contribute of the previous steps ($\mu_o(i,j-1)p(i,j-1)$, $\mu_o(i-1,j-1)p(i-1,j-1)$ and $\mu_o(i-1,j)p(i-1,j)$) the number of gap openings in the current step. It is easy to verify that it is given by $p(i-1,j) - p(i-2,j) + p(i,j-1) - p(i,j-2)$.

Algorithm 1. Computation of μ_m

Input: a pair of sequences S_1 and S_2.

$p(0,0) = 1,\ \mu_m(0,0) = 0$
`for` $i = 1 : n_1$,
    $p(i,0) = 1$
    $\mu_m(i,0) = 0$
`end`
`for` $j = 1 : n_2$
    $p(0,j) = 1$
    $\mu_m(0,j) = 0$
`end`
`for` $i = 1 : n_1$
    `for` $j = 1 : n_2$
        $p(i,j) = p(i-1,j-1) + p(i,j-1) + p(i-1,j)$
        `if` $s_1(i) = s_2(j)$ `then` $M = 1$ `else` $M = 0$
        $\mu_m(i,j) = \frac{\mu_m(i-1,j)p(i-1,j)+\mu_m(i,j-1)p(i,j-1)+(\mu_m(i-1,j-1)+M)p(i-1,j-1)}{p(i,j)}$
    `end`
`end`
Output: $\mu_m(n_1, n_2)$

Extension to a General Scoring Matrix. Assuming to align sequences of amino acids we have different parameters for each pair of them (in total 210 parameters) plus other 2 parameters for gap opening and gap extension. The

formulas developed above apply with minor modifications. Concerning the mean values, μ_o and μ_e are calculated as before. For the others it is:

$$\mu_{q_{ty}}(i,j) = \frac{1}{p(i,j)}(\mu_{q_{ty}}(i-1,j)p(i-1,j) + \mu_{q_{ty}}(i,j-1)p(i,j-1) \\ +(\mu_{q_{ty}}(i-1,j-1) + M)p(i-1,j-1))$$

where $M = 1$ when $s'_1(i) = t$ and $s'_2(j) = y$ or viceversa with $t, y \in \mathcal{S}$. The matrix $\boldsymbol{C}$ is a symmetric matrix 212x212. The values v_{eo}, v_{ee} and v_{oo} are calculated as above. The derivation of formulas for $v_{q_{ty}q_{t'y'}}$ is straightforward from v_{ms} considering the appropriate values for M and the mean values. The recursions for $v_{oq_{ty}}$ and $v_{eq_{ty}}$ are similar to those of v_{mo} and v_{me}.

4 Inverse Parametric Sequence Alignment Problem

In IPSAP the task is to determine a value for the scoring parameters that makes a given set of aligned sequence pairs optimal. Let consider as training set m pairs of sequences S_{1i} and S_{2i}, $i = 1 \dots m$ and their optimal alignments $\bar{\mathcal{A}}_i$. We are interested in determining values of the parameters $\boldsymbol{\alpha}$ such that:

$$\phi\left(S_{1i}, S_{2i}, \bar{\mathcal{A}}_i\right) \geq \phi\left(S_{1i}, S_{2i}, \mathcal{A}_{ij}\right) \quad \forall \mathcal{A}_{ij} \neq \bar{\mathcal{A}}_i \quad \forall\, i = 1 \dots m \quad \forall\, j = 1 \dots N$$

Due to the linearity of $\phi\left(S_{1i}, S_{2i}, \bar{\mathcal{A}}_i\right)$ these constraints can be expressed as a set of linear inequality constraints $\mathcal{C}$:

$$\boldsymbol{\alpha}^T \bar{\mathbf{x}}_i \geq \boldsymbol{\alpha}^T \mathbf{x}_{ij} \quad \forall \mathcal{A}_{ij} \neq \bar{\mathcal{A}}_i \quad \forall\, i = 1 \dots m \quad \forall\, j = 1 \dots N \tag{3}$$

These constraints specify a convex set to which the parameters may belong. A common approach [6,7] is to define an objective function, which is minimized subject to these constraints. In this paper, we choose the objective function in a deliberate way, the aim being twofold. First, as in previous approaches it is easy to optimize (i.e., convex). Second, it is designed such that most, if not all, of the constraints are satisfied automatically, further reducing computational cost.

Such an objective function is given by the Z-score. In fact, given a pair of aligned sequences, scoring parameter vectors with a high Z-score correspond to few alignments with score higher than that of the given one. On the contrary a small Z-score indicates a low position of the target alignment in the ranking associated with that scoring model. Interestingly, under normality assumptions (which we verified to be close to valid for the score distributions examined in the experiments) this Z-score is directly equivalent to a p-value. Hence, maximizing the Z-score can be interpreted as maximizing the significance of the correct alignment score: the larger the Z-score, the more significant the correct alignment is, or the more different it is from the distribution of random alignments.

The problem we are interested in is:

$$\max_{\boldsymbol{\alpha}} \frac{\boldsymbol{\alpha}^T \boldsymbol{b}^*}{\sqrt{\boldsymbol{\alpha}^T \boldsymbol{C}^* \boldsymbol{\alpha}}}$$

Since the objective is invariant to a positive scaling and due to the monotonicity of the square root, we can reformulate the problem as a quadratic programming (QP) problem:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \ \boldsymbol{\alpha}^T \boldsymbol{C}^* \boldsymbol{\alpha} \\ &\text{s.t.} \quad \boldsymbol{\alpha}^T \boldsymbol{b}^* \geq 1 \end{aligned} \tag{4}$$

Note that being $\boldsymbol{C}^*$ positive definite, the objective function is convex.

The result of using the Z-score is that the constraints are generally inactive, as directly maximizing the Z-score without any constraints (3) often leads to a result that automatically satisfies them all. Naturally, in some cases the result of (4) still violates some of them, and one may want to impose these explicitly. This feasible region is also convex, hence the optimization problem remains convex, and solvable by standard means. To solve this problem one can adopt an incremental algorithm that starts without any constraints (3) and progressively add the most violated constraint until convergence is reached (see Algorithm 2 in Appendix). The algorithm is then of a similar structure as previous approaches to IPSAP such as [7]. However, the crucial difference is that we use another cost function, which ensures that the number of iterations is generally much smaller.

Note that the unconstrained method based on the Z-score (Eq. 4) elegantly deals with cases where there exists no parameter setting for which the given alignments are optimal. On the other hand, the constraint-based approach would have an infeasible constraint set in this case. This problem can be solved by relaxing the constraints, by requiring the inequalities to hold subject to a small tolerance ϵ (as in [6]).

Algorithm 2. Incremental algorithm to incorporate active constraints.

Input: m optimal alignments $\bar{\mathcal{A}}_i$ between S_{1i} and S_{2i}. The required accuracy ϵ.

`for` $i = 1 \ldots m$ compute $\boldsymbol{b}_i$ and $\boldsymbol{C}_i$
Compute $\boldsymbol{b}^* := sum(\boldsymbol{b}_i)$ and $\boldsymbol{C}^* := sum(\boldsymbol{C}_i)$
Find $\boldsymbol{\alpha}_{opt}$ solving QP (4)
`Repeat`
  $exit := 0$
  `for` $i = 1 \ldots m$
    Compute $\mathbf{x}'_i := \arg\max_{\mathbf{x}} \phi\,(S_{1i}, S_{2i}, \mathcal{A}_i)$
    `If` $\boldsymbol{\alpha}_{opt}^T \mathbf{x}'_i - (1+\epsilon)\boldsymbol{\alpha}_{opt}^T \bar{\mathbf{x}} \geq 0$
      $exit := 1$
      $\mathcal{C} := \mathcal{C} \cup \{\boldsymbol{\alpha}^T((1+\epsilon)\bar{\mathbf{x}} - \mathbf{x}'_i) \geq 0\}$
      Find $\boldsymbol{\alpha}_{opt}$ solving QP (4) s.t. $\mathcal{C}$
  `end`
`until` $exit = 1$
Output: $\boldsymbol{\alpha}_{opt}$

5 Experimental Results

In this section we present some results obtained by applying our method to both real and artificial amino acids sequences. The first series of experiments shows the effectiveness of our approach for an arbitrary number of parameters when the optimization problem QP is feasible. Artificial pairs of sequences of length 100 have been randomly generated and split into training and test sets. The size of the training set is variable while the test set is fixed to 100 pairs. A random vector of parameters is considered to align the sequences. Then our algorithm is used to recover the optimal values of the scoring parameters and the original sequences are realigned. We measure the performance of our method in terms of correctly recovered alignments. While the error on the training set is always zero, Fig. 1.a depicts the average test error on 1000 different experiments. As expected the learning curves approaches to zero still for training set of small dimensions and the convergence is faster for the model with three parameters.

We also consider the number of constraints that must be incorporated before the algorithm converges in the three parameter model. Again 1000 pairs of random sequences of length 100 have been generated and aligned with a random set of parameters. Then our algorithms is used to calculate the optimal parameters for each pair of sequences separately. The number of constraints needed to reach convergence is measured. Figure 1.b shows that on average 85% of pairs are correctly aligned solving the QP without incorporating any constraints, about 14% of pairs needs one or two constraints, while the maximum number of constraints does not exceed three.

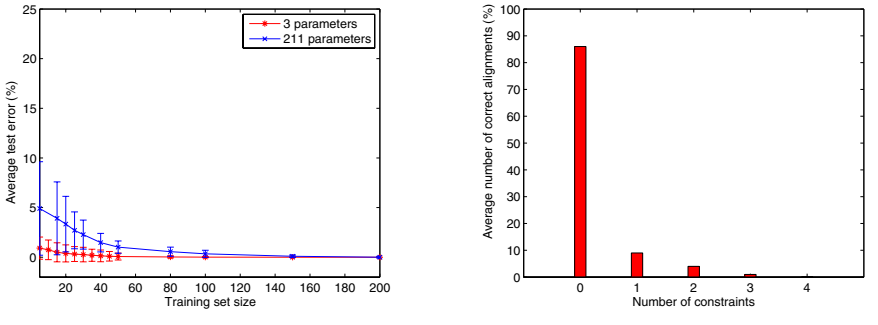


Fig. 1. (a) Test error as function of the training set size. (b) Distribution of correctly reconstructed alignments as a function of the number of additional constraints.

Finally we assess the effectiveness of the proposed method on real sequences of amino acids. We perform experiments similar to those presented in the paper of Kececioglu *et al.* [6]. We consider five multiple alignments from the PALI database [1] of structural protein alignments: T-boxes (`box`), NADH oxidoreductases (`nad`), Kunitz inhibitors (`kun`), Sir2 transcription regulators (`sir`) and

pectin methylesterases (pec). For each group of proteins we selected ten sequences as training set and ten for the test set.

Two different models of scoring parameters are evaluated, with 4 and 212 parameters respectively. Since the problems are infeasible a tolerance constant ϵ must be set to fix the required accuracy: it can be done with a fast binary search strategy as in [6]. Results are shown in Table 1. Here performances for both models are evaluated in terms of the number of alignments that scores higher than the given ones (alignments with the same summary $\mathbf{x}$ are considered the same). As shown in the table this number is quite low for both models. To have an idea we should keep in mind that the number of possible optimal alignment summaries is bounded from above by a polynomial in n_1 and n_2 of degree $\frac{k(k-1)}{k+1}$, where k is the number of the parameters [9]. This clearly shows that the parameters vector given by our algorithm is a near-optimal solution. Also the number of constraints added is quite low (< 45): it means that less than one constraint for training pair is needed.

Table 1. Error rates and added constraints (in parenthesis) in the PALI dataset

	4 parameters		212 parameters	
Dataset	Tr error	Te error	Tr error	Te error
nad	4.95 (5)	6.46	567.46 (21)	703.12
kun	1.46 (12)	0.95	386.46 (41)	457.3
box	1 (3)	1.13	211.3 (12)	256.7
sir	1 (10)	1.16	236 (36)	301.44
pec	46.2 (8)	76.1	835.12 (31)	1054.12

6 Conclusions

We developed a new algorithm for IPSAP, which efficiently finds a scoring parameter setting consistent with a given training set of correct alignments. The algorithm is fast and easy to implement since it relies on dynamic programming to compute the Z-score as a function of the parameters, and a very simple quadratic program to subsequently maximize the Z-score. It can be applied to an arbitrary number of parameters provided appropriate recursive relations for mean values and covariance matrices are used. It naturally and adequately deals with the infeasible case where there exists no parameter setting for which the correct alignments are optimal.

Compared to previous methods, the proposed approach has the benefit that no or only very few constraints need to be added, such that in practice is clearly faster for long sequences. The cost of computing $\boldsymbol{b}$ and $\boldsymbol{C}$ is fixed ($O(kn_1n_2)$ where k is the number of parameters) and is performed only once at the beginning of the method, whereas in other approaches many times the NW algorithm needs

to be invoked. Furthermore, it is conceivable that approximate algorithms can be developed to obtain an estimate of the mean and variance (e.g. a method based on random walks along the possible paths) with a significantly reduced computational cost.

Aknowledgments

This work was partially supported by NIH grant R33HG003070-01, and the EU Project SMART.

References

1. Balaji, S., Sujatha, S., Kumar, S.S.C., Srinivasan, N.: PALI: a database of alignments and phylogeny of homologous protein structures. Nucleic Acids Research 29, 1, 61–61 (2001)
2. Eppstein, D.: Setting parameters by example. In: ACM Computing Research Repository, 40th IEEE Symp. Foundations of Comp. Sci., pp. 309–318 (1999), SIAM J. Computing 32(3), 643–653 (2003)
3. Goldberg, M., Breimer, E.: Learning Significant Alignments: An Alternative to Normalized Local Alignment. In: Proceedings of the International Symposium on Methodologies for Intelligent Systems, pp. 37–45 (2002)
4. Gusfield, D., Balasubramanian, K., Naor, D.: Parametric optimization of sequence alignment. Algorithmica 12, 312–326 (1994)
5. Gusfield, D., Stelling, P.: Parametric and inverse-parametric sequence alignment with XPARAL. Methods in Enzymology 266, 481–494 (1996)
6. Kececioglu, J., Kim, E.: Simple and fast inverse alignment. In: Proc. of the 10th ACM Conference on Research in Computational Molecular Biology, pp. 441–455 (2006)
7. Joachims, T., Galor, T., Elber, R.: Learning to Align Sequences: A Maximum-Margin Approach. In: Leimkuhler, B. (ed.) New Algorithms for Macromolecular Simulation. LNCSE, vol. 49, Springer, Heidelberg (2005)
8. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. 48, 443–453 (1970)
9. Pachter, L., Sturmfels, B.: Parametric inference for biological sequence analysis. Proceedings of the National Academy of Sciences USA 101, 46, 16138–16143 (2004)
10. Sun, F., Fernandez-Baca, D., Yu, W.: Inverse parametric sequence alignment. Journal of Algorithms 53, 36–54 (2004)

Appendix

Algorithm 3. Extra recursions for the three and four parameter models

$$\mu_s(i,j) = \tfrac{1}{p(i,j)}(\mu_s(i-1,j)p(i-1,j) + \mu_s(i,j-1)p(i,j-1) \\ +(\mu_s(i-1,j-1) + (1-M))p(i-1,j-1))$$

$$\mu_g(i,j) = \tfrac{1}{p(i,j)}(\mu_g(i-1,j) + 1)p(i-1,j) + (\mu_g(i,j-1) + 1)p(i,j-1) \\ +\mu_g(i-1,j-1)p(i-1,j-1))$$

$$\mu_e(i,j) = \tfrac{1}{p(i,j)}(\mu_e(i-1,j)p(i-1,j) + p(i-2,j) + \mu_e(i,j-1)p(i,j-1) \\ +p(i,j-2) + \mu_e(i-1,j-1)p(i-1,j-1))$$

$$\mu_o(i,j) = \tfrac{1}{p(i,j)}(\mu_o(i-1,j)p(i-1,j) + p(i-1,j) - p(i-2,j)+ \\ \mu_o(i,j-1)p(i,j-1) + p(i,j-1) - p(i,j-2) + \mu_o(i-1,j-1)p(i-1,j-1))$$

$$v_{mm}(i,j) = \tfrac{1}{p(i,j)}(v_{mm}(i-1,j)p(i-1,j) + v_{mm}(i,j-1)p(i,j-1) \\ +(v_{mm}(i-1,j-1) + 2M\mu_m(i-1,j-1) + M)p(i-1,j-1))$$

$$v_{ss}(i,j) = \tfrac{1}{p(i,j)}(v_{ss}(i-1,j)p(i-1,j) + v_{ss}(i,j-1)p(i,j-1) \\ +(v_{ss}(i-1,j-1) + 2(1-M)\mu_s(i-1,j-1) + (1-M))p(i-1,j-1))$$

$$v_{gg}(i,j) = \tfrac{1}{p(i,j)}(v_{gg}(i-1,j) + 2\mu_g(i-1,j) + 1)p(i-1,j) \\ +(v_{gg}(i,j-1) + 2\mu_g(i,j-1) + 1)p(i,j-1) + v_{gg}(i-1,j-1)p(i-1,j-1))$$

$$v_{mg}(i,j) = \tfrac{1}{p(i,j)}(v_{mg}(i-1,j) + \mu_m(i-1,j))p(i-1,j) + (v_{mg}(i,j-1) \\ +\mu_m(i,j-1))p(i,j-1) + (v_{mg}(i-1,j-1) + M\mu_g(i-1,j-1))p(i-1,j-1))$$

$$v_{sg}(i,j) = \tfrac{1}{p(i,j)}(v_{sg}(i-1,j) + \mu_s(i-1,j))p(i-1,j) + (v_{sg}(i,j-1) \\ +\mu_s(i,j-1))p(i,j-1) + (v_{sg}(i-1,j-1) \\ +(1-M)\mu_g(i-1,j-1))p(i-1,j-1))$$

$$
\begin{aligned}
v_{ms}(i,j) = \tfrac{1}{p(i,j)}(&v_{ms}(i-1,j)p(i-1,j) + v_{ms}(i,j-1)p(i,j-1) \\
&+(v_{ms}(i-1,j-1) + M\mu_s(i-1,j-1) + (1-M)\mu_m(i-1,j-1))p(i-1,j-1))
\end{aligned}
$$

$$
\begin{aligned}
v_{oo}(i,j) = \tfrac{1}{p(i,j)}(&v_{oo}(i-1,j-1)p(i-1,j-1) + v_{oo}(i-1,j)p(i-1,j) \\
&+2(\mu_o(i-1,j)p(i-1,j) - \mu_o(i-2,j)p(i-2,j) - p(i-2,j) + p(i-3,j)) \\
&+p(i-1,j) - p(i-2,j) + v_{oo}(i,j-1)p(i,j-1) + 2(\mu_o(i,j-1)p(i,j-1) \\
&-\mu_o(i,j-2)p(i,j-2) - p(i,j-2) + p(i,j-3)) + p(i,j-1) - p(i,j-2)
\end{aligned}
$$

$$
\begin{aligned}
v_{ee}(i,j) = \tfrac{1}{p(i,j)}(&v_{ee}(i-1,j-1)p(i-1,j-1) + v_{ee}(i-1,j)p(i-1,j) \\
&+2\mu_e(i-2,j)p(i-2,j) + 2p(i-3,j) + p(i-2,j) + v_{ee}(i,j-1)p(i,j-1) \\
&+2\mu_e(i,j-2)p(i,j-2) + 2p(i,j-3) + p(i,j-2))
\end{aligned}
$$

$$
\begin{aligned}
v_{mo}(i,j) = \tfrac{1}{p(i,j)}(&(v_{mo}(i-1,j) + \mu_m(i-1,j))p(i-1,j) - \mu_m(i-2,j)p(i-2,j) \\
&+(v_{mo}(i,j-1) + \mu_m(i,j-1))p(i,j-1) - \mu_m(i,j-2)p(i,j-2) \\
&+(v_{mo}(i-1,j-1) + M\mu_o(i-1,j-1))p(i-1,j-1))
\end{aligned}
$$

$$
\begin{aligned}
v_{me}(i,j) = \tfrac{1}{p(i,j)}(&(v_{me}(i-1,j-1) + M\mu_e(i-1,j-1))p(i-1,j-1) \\
&+v_{me}(i-1,j)p(i-1,j) + \mu_m(i-2,j)p(i-2,j) + v_{me}(i,j-1)p(i,j-1) \\
&+\mu_e(i,j-2)p(i,j-2))
\end{aligned}
$$

$$
\begin{aligned}
v_{eo}(i,j) = \tfrac{1}{p(i,j)}(&v_{eo}(i-1,j-1)p(i-1,j-1) + v_{eo}(i,j-1)p(i,j-1) \\
&+\mu_o(i,j-2)p(i,j-2) + p(i,j-2) - 2p(i,j-3) + \mu_e(i,j-1)p(i,j-1) \\
&-\mu_e(i,j-2)p(i,j-2) + v_{eo}(i-1,j)p(i-1,j) + \mu_o(i-2,j)p(i-2,j) \\
&+p(i-2,j) - 2p(i-3,j) + \mu_e(i-1,j)p(i-1,j) - \mu_e(i-2,j)p(i-2,j))
\end{aligned}
$$

Transductive Reliability Estimation for Kernel Based Classifiers

Dimitris Tzikas[1], Matjaz Kukar[2], and Aristidis Likas[1]

[1] Department of Computer Science, University of Ioannina, Greece
[2] Faculty of Computer and Information Science, University of Ljubljana, Slovenia
tzikas@cs.uoi.gr, matjaz.kukar@fri.uni-lj.si, arly@cs.uoi.gr

Abstract. Estimating the reliability of individual classifications is very important in several applications such as medical diagnosis. Recently, the transductive approach to reliability estimation has been proved to be very efficient when used with several machine learning classifiers, such as Naive Bayes and decision trees. However, the efficiency of the transductive approach for state-of-the art kernel-based classifiers was not considered. In this work we deal with this problem and apply the transductive reliability methodology with sparse kernel classifiers, specifically the Support Vector Machine and Relevance Vector Machine. Experiments with medical and bioinformatics datasets demonstrate better performance of the transductive approach for reliability estimation compared to reliability measures obtained directly from the output of the classifiers. Furthermore, we apply the methodology in the problem of reliable diagnostics of the coronary artery disease, outperforming the expert physicians' standard approach.

1 Introduction

Decision-making is usually an uncertain and complicated process, therefore it is often crucial to know the magnitude of diagnosis' (un)reliability in order to minimize risks, for example in the medical domain risks related to the patient's health or even life. One of the reasons why machine learning methods are infrequently used in practice is that they fail to provide an unbiased reliability measure of predictions.

Although there are several methods for estimating the overall performance of a classifier, e.g cross-validation, and quality (reliability and validity) of collected data [1], there is very little work on estimating the reliability of individual classifications. The transductive reliability methodology as introduced in [2] computes the reliability of an individual classification, by studying the stability of the trained model when the training set is perturbed (the newly classified example is added to the training set and the classifier is retrained). For reliable classifications, this process should not lead to significant model changes. The transductive reliability methodology has been applied on traditional classifiers like Naive Bayes and decision trees with interesting results. Here, we examine the effectiveness of this methodology when applied on sparse kernel-based classifiers, such as the Support Vector Machine (SVM) and the Relevance Vector

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 37–47, 2007.

Machine (RVM), and compare transductive reliability estimations with reliability measures based on the outputs that SVM and RVM provide. Furthermore, we apply the methodology for diagnosis of the coronary artery disease (CAD) using kernel-based classifiers and compare our results to the performance of expert physicians using an established standard methodology.

2 Transduction Reliability Estimations

Transduction is an inference principle that takes a training sample and aims at estimating the values of a discrete or continuous function only at given unlabeled points of interest from input space, as opposed to the whole input space for induction. In the learning process the unlabeled points are suitably labelled and included into the training sample. The usefullness of unlabeled data has also been advocated in the context of co-training. It has been shown [3] that for every better-than-random classifier its performance can be significantly improved by utilizing only additional unlabeled data.

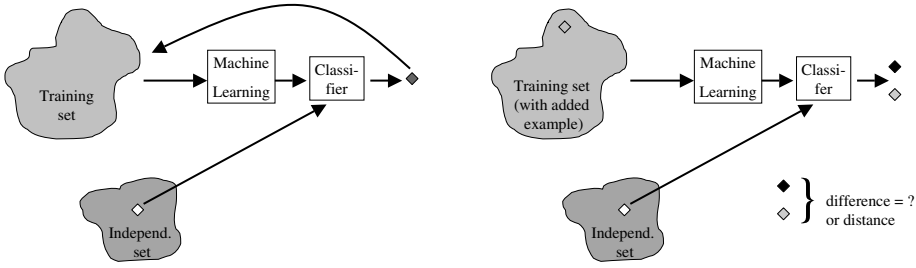
The transductive reliability estimation process and its theoretical foundations originating from Kolmogorov complexity are described in more detail in [2]. In practice, it is performed in a two-step process, featuring an *inductive step* followed by a *transductive step*.

- An *inductive step* is just like an ordinary inductive learning process in Machine Learning. A Machine Learning algorithm is run on the training set, *inducing* a classifier. A selected example is taken from an independent dataset and classified using the induced classifier. An example, labelled with the predicted class is temporarily included into the training set (Figure 1a).
- A *transductive step* is almost a repetition of an inductive step. A Machine Learning algorithm is run on the changed training set, *transducing* a classifier. The same example as before is taken from the independent dataset and is classified using the transduced classifier (Figure 1b). Both classifications of the same example are compared and their difference (distance) is calculated, thus approximating the randomness deficiency.
- After the reliability is calculated, the example in question is removed from the training set.

The machine learning algorithm, whose reliability is being assessed, is assumed to provide a probability distribution p that describes the probability that its input belongs at each possible class. In order to measure how much the model changes, we calculate the distance between the probability distribution p of the initial classifier and the probability distribution q of the augmented classifier, using the Symmetric Kullback-Leibler divergence, or J-divergence, which is defined as

$$J(p,q) = \sum_{i=1}^{n} (p_i - q_i) \log_2 \frac{p_i}{q_i}. \tag{1}$$

$J(p,q)$ is limited to the interval $[0,\infty)$, with $J(P,P) = 0$. For the ease of interpretation, it is desirable for reliability values to be bounded to the $[0,1]$ interval, $J(p,q)$ is normalized in the spirit of Martin-Löf's test for randomness



(a) Inductive step. (b) Transductive step.

Fig. 1. Transductive reliability estimation

[2,4, pp. 129], to obtain the transductive reliability measure (TRE) used in our approach:

$$TRE = 1 - 2^{-J(p,q)}. \tag{2}$$

Due to non-optimal classifiers resulting from learning in noisy and incomplete datasets, it is inappropriate to select *a priori* fixed boundary (say, 0.90) as a threshold above which a classification is considered reliable. To deal with this problem, we split the range $[0, 1]$ of reliability estimation values into two intervals by selecting a threshold T. The lower interval $[0, T)$ contains unreliable classifications, while the higher interval $[T, 1]$ contains reliable classifications. As a splitting point selection criterion we use maximization of the information gain [5]:

$$Gain = H(S) - \frac{|S_1|}{|S|}H(S_1) - \frac{|S_2|}{|S|}H(S_2), \tag{3}$$

where $H(S)$ denotes the entropy of set S, $S_1 = \{x : TRE(x) < T\}$ is the set of unreliable examples and $S_2 = \{x : TRE(x) > T\}$ is the set of reliable results.

Note that our approach is considerably different from that described in [6,7]. Their approach is tailor-made for SVM (it works by manipulating support vectors) while ours requires only that the applied classifier provide a probability distribution. Our approach can also be used in conjunction with probability calibration, e.g. by utilizing the typicalness concept [8,9].

3 Kernel-Based Methods

Kernel methods have been extensively used to solve classification problems, where a training set $\{x_n, t_n\}_{n=1}^{N}$ is given, so that t_n denotes the class label of training example x_n. The class labels t_n are discrete, e.g. $t \in \{0, 1\}$ for binary classification, and they describe the class to which each training example belongs. Kernel methods, are based on a mapping function $\phi(x)$ that maps each training vector to a higher dimensional feature space. Then, inner products between training examples are computed in this new feature space, by evaluating

the corresponding kernel function $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$. This kernel function, provides a measure of similarity between training examples.

Recently, there is much interest in sparse kernel methods, such as the Support Vector Machine (SVM) and the Relevance Vector Machine (RVM). These methods are called sparse because, after training with the full dataset, they make predictions using only a small subset of the available training vectors. In SVM sparsity is achieved through suitable weight regularization, while RVM is a Bayesian model and sparsity is a consequence of the use of a suitable sparse prior distribution on the weights. The remaining training vectors, which are used for predictions are called support vectors (SV) in the case of SVM and relevance vectors (RV) in the case of RVM. The main reason why sparse kernel methods are so interesting and effective, is that during training, they automatically estimate the complexity of the dataset, and thus they have good generalization performance on both simple and complex datasets. In simple datasets only few support/relevance vectors will be used, while in more difficult datasets the number of support/relevance vectors will increase. Furthermore, making predictions using only a small subset of the initial training examples is typically much more computationally efficient.

3.1 Support Vector Machine

The support vector machine (SVM) classifier, is a kernel classifier that aims at finding an optimal hyperplane which separates data points of two classes. This hyperplane is optimal in the sense that it maximizes the margin between the hyperplane and the training examples. The SVM classifier [10] makes decisions for an unknown input vector, based on the sign of the decision function:

$$y_{SVM}(x) = \sum_{n=1}^{N} w_n K(x, x_n) + b \tag{4}$$

After training, most of the weights w are set to exactly zero, thus predictions are made using only few of the training vectors, which are the support vectors. Assuming that the two classes are labeled with '-1' and '1', so that $t_n \in \{-1, 1\}$, the weights $w = (w_1, \ldots, w_N)$ are set by solving the following quadratic programming problem:

$$\begin{aligned} \min_{w,b,\xi} \ & \frac{1}{2} w^T w + C \sum_{n=1}^{N} \xi_n \\ \text{subject to } \ & t_n(w^T \phi(x_n) + b) \geq 1 - \xi_n \\ & \xi_n \geq 0 \end{aligned} \tag{5}$$

where the auxiliary variables $\xi = (\xi_1, \ldots, \xi_N)$ have been introduced to deal with non-separable datasets.

SVM makes predictions based on the decision function of eq. (4). Positive values of the decision function ($y_{SVM}(x) > 0$) correspond to class '1', while negative values ($y_{SVM}(x) < 0$) correspond to class '-1'. Furthermore, the absolute

value of the decision function provides a measure of the certainty of the decision. Values near zero, correspond to points near the decision boundary and therefore may be unreliable, while large values of the decision function should correspond to reliable classifications. In practice, we first obtain probabilistic predictions by applying the sigmoid function $\sigma(x) = 1/(1+\exp(-x))$ to the SVM outputs and then compute the reliability measure as:

$$RE_{SVM} = |2\sigma(y_{SVM}(x)) - 1|. \tag{6}$$

3.2 Relevance Vector Machine

The relevance vector machine (RVM) classifier [11], is a probabilistic extension of the linear regression model, which provides sparse solutions. It is analogous to the SVM, since it computes the decision function using only few of the training examples, which are now called relevance vectors. However training is based on different objectives.

The RVM model $y(x; w)$ is the output of a linear model with parameters $w = (w_1, \ldots, w_N)^T$, with application of a sigmoid function for the case of classification:

$$y_{RVM}(x) = \sigma(\sum_{n=1}^{N} w_n K(x, x_n)), \tag{7}$$

where $\sigma(x) = 1/(1+\exp(-x))$. In the RVM, sparseness is achieved by assuming a suitable prior distribution on the weights, specifically a zero-mean, Gaussian distribution with distinct inverse variance α_n for each weight w_n:

$$p(w|\alpha) = \prod_{n=1}^{N} N(w_n|0, \alpha_n^{-1}). \tag{8}$$

The variance hyperparameters $\alpha = (\alpha_1, \ldots, \alpha_N)$ are assumed to be Gamma distributed random variables:

$$p(\alpha) = \prod_{n=1}^{N} Gamma(\alpha_n|a, b). \tag{9}$$

The parameters a and b are assumed fixed and usually they are set to zero ($a = b = 0$), which provides sparse solutions.

Given a training set $\{x_n, t_n\}_{n=1}^{N}$ with $t_n \in \{0, 1\}$ training in RVM is equivalent to compute the posterior distribution $p(w, \alpha|t)$. However, since this computation is intractable, a quadratic approximation $\log p(w|t, \alpha) \approx (w-\mu)^T \Sigma^{-1}(w-\mu)$ is assumed and we compute matrix Σ and vector μ as:

$$\Sigma = (\Phi^T B \Phi + A)^{-1} \tag{10}$$

$$\mu = \Sigma \Phi^T B \hat{t} \tag{11}$$

with the $N \times N$ matrix Φ defined as $[\Phi]_{ij} = K(x_i, x_j)$, $A = \mathrm{diag}(\alpha_1, \ldots, \alpha_N)$, $B = \mathrm{diag}(\beta_1, \ldots, \beta_N)$, $\beta_n = y_{RVM}(x_n)[1 - y_{RVM}(x_n)]$ and $\hat{t} = \Phi\mu + B^{-1}(t - y)$.

The parameters α are then set to the values α_{MP} that maximize the logarithm of the following marginal likelihood

$$L(\alpha) = \log p(\alpha|t) = -\frac{1}{2}\left[N \log 2\pi + log|C| + t^T C^{-1} t\right], \tag{12}$$

with $C = B^{-1} + \Phi A^{-1} \Phi^T$. This, gives the following update formula:

$$\alpha_n = \frac{1 - \alpha_n \Sigma_{nn}}{\mu_n^2} \tag{13}$$

The RVM learning algorithm iteratively evaluates formulas (10),(11) and (13).

After training, the value of $y_{RVM}(x) = y(x; \mu)$ can be used to estimate the reliability of the classification decision for input x. Values close to 0.5 are near the decision boundary and therefore are unreliable classifications, while values near 0 and near 1 should correspond to reliable classifications. In our experiments, we used the reliability measure

$$RE_{RVM} = |2y_{RVM}(x) - 1|, \tag{14}$$

which takes values near 0 for unreliable classifications and near 1 for reliable classifications.

3.3 Incremental Relevance Vector Machine

An interesting property of the RVM model that can be exploited in the transductive approach, is that it can be trained incrementally, as proposed in [12]. The proposed incremental algorithm, initially assumes an empty model, that does not use any basis functions. Then, it incrementally adds, deletes and re-estimates basis functions, until convergence. It is based on the observation that the marginal likelihood, see eq. (12), can be decomposed as:

$$L(\alpha) = L(\alpha_{-n}) + l(\alpha_n), \tag{15}$$

where $L(\alpha_{-n})$ does not depend on α_n and

$$l(\alpha_n) = \log \alpha_n - \log(\alpha_n + s_n) + \frac{q_n^2}{\alpha_n + s_n}, \tag{16}$$

with $s_n = \phi_n^T C_{-n}^{-1} \phi_n$ and $q_n = \phi_n^T C_{-n}^{-1} \hat{t}$. Here, $C_{-n} = B^{-1} + \sum_{i \neq n} \alpha_n^{-1} \phi_n \phi_n^T$ denotes the matrix C without the contribution of the n-th basis function, so that $C = C_{-n} + \alpha_n^{-1} \phi_n \phi_n^T$, s_n is the "sparseness" factor that measures how sparse the model is and q_n is the "quality" factor that measures how well the model fits the observations. Based on this decomposition, analysis of $l(\alpha_n)$ shows that it is maximized when

$$\alpha_n = \frac{s_n^2}{q_n^2 - s_n} \qquad \text{if } q_n^2 > s_n \tag{17}$$

$$\alpha_n = \infty \qquad \text{if } q_n^2 \leq s_n \tag{18}$$

Based on this result, the following algorithm is proposed in [12]:

1. Initially assume an empty model, set $a_n = \infty$, for all n
2. Select a training point x_n and compute the corresponding basis function ϕ_n as well as s_n and q_n.
   (a) if $q_n^2 > s_n$ and $\alpha_n = \infty$ add the basis function to the model, using eq. (17) to set α_n
   (b) if $q_n^2 > s_n$ and $\alpha_n < \infty$ re-estimate α_n
   (c) if $q_n^2 <= s_n$ remove the basis function from the model, set $\alpha_n = \infty$
3. Compute Σ and μ, using eq. (10) and (11)
4. Repeat from step 2, until convergence.

4 Evaluation of Transductive Reliability Estimations

In this section, we apply the transductive reliability methodology in a series of classification problems and compare the performance of transductive reliability estimations, with respect to the reliability measures that are directly computed based on SVM and RVM outputs. Transductive reliability estimations, are obtained following the procedure described in Section 2. After training the model and computing its output for a new test point x_*, we add this test point to the training set with the predicted label and retrain the model. Transductive reliability estimations are obtained by measuring the distance between the output distributions of the two models.

In the case of RVM we also considered a modification, where we used the incremental training algorithm to obtain fast transductive reliability estimations. Specifically, after adding the new training point x_*, instead of retraining from scratch, we can use the incremental algorithm to continue training the previous model. This is much more computationally efficient, and in the experiments it appears to provide better performance than the standard approach of training from scratch.

In order to evaluate the performance of the reliability estimation methods, we apply the following procedure. We perform leave-one-out cross-validation on the available training dataset and compute a prediction for the class of each training point and a reliability estimation (RE) of this prediction. Afterwards, we can discriminate reliable and unreliable classifications by selecting a threshold (T)

Table 1. Information gain of SVM/RVM reliability estimations and transductive reliability estimations

Method	hepatitis	new-thyroid	wdbc	leukemia
RE_{SVM}	0.106	0.083	0.036	0.054
TRE_{SVM}	0.120	0.092	0.047	0.073
RE_{RVM}	0.109	0.068	0.091	0.089
TRE_{RVM}	0.178	0.062	0.094	0.062
$TRE_{RVM(inc)}$	0.133	0.072	0.106	0.107

for the reliability measure. Using an ideal reliability measure all correct classifications should be labeled reliable ($RE > T$), while all incorrect classifications should be labeled unreliable. Thus, an evaluation of the reliability measure is obtained by computing the percentage of correct and reliable classifications, and the percentage of incorrect reliable classifications. Plotting these percentages, for many values of the threshold, produces an ROC curve, which illustrates the performance of the reliability estimation method.

Although the ROC describes the overall effectiveness of a reliability measure, in practice, a single threshold value has to be used. This is selected by maximizing the information gain, as explained in Section 2. The information gain may also be used to compare the performance of several reliability measures. Table 1 shows the information gain that is achieved by: i) using directly the SVM/RVM reliability estimates RE_{SVM} and RE_{RVM}, ii) using the transduction reliability principle (TRE). Results are shown for three medical datasets from the UCI machine learning repository and the leukemia bioinformatics dataset. It is clear that when SVM is used, transduction provides better information gain for all datasets. The same happens with incremental RVM, while when typical RVM is used, transduction is better in two of the three cases.

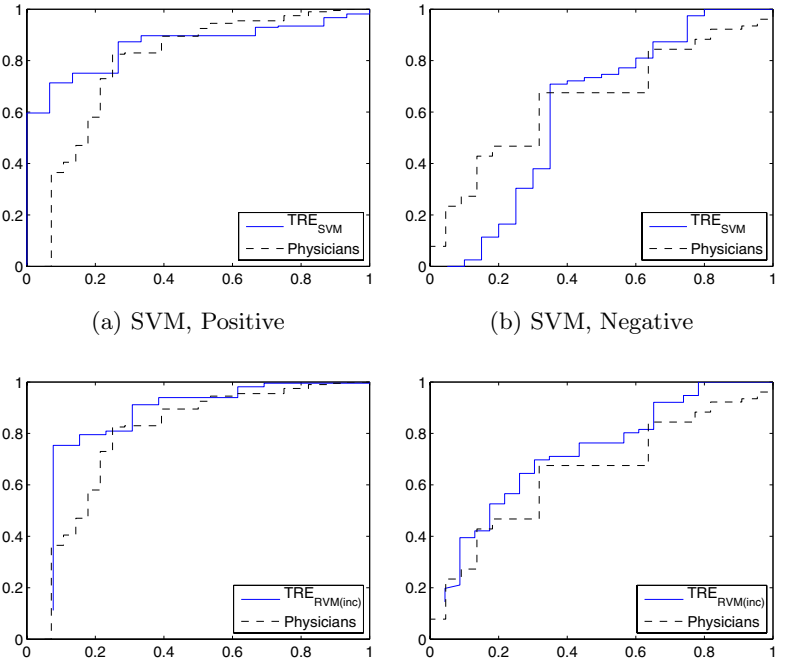
5 Diagnosis of Coronary Artery Disease

Coronary artery disease (CAD) is the most important cause of mortality in all developed countries. It is caused by diminished blood flow through coronary arteries due to stenosis or occlusion. CAD produces impaired function of the heart and finally the necrosis of the myocardium – myocardial infarction.

In our study we used a dataset of 327 patients (250 males, 77 females) with performed clinical and laboratory examinations, exercise ECG, myocardial scintigraphy and coronary angiography because of suspected CAD. The features from the ECG an scintigraphy data were extracted manually by the clinicians. In 228 cases the disease was angiographically confirmed and in 99 cases it was excluded. 162 patients had suffered from recent myocardial infarction. The patients were selected from a population of approximately 4000 patients who were examined at the Nuclear Medicine Department, University Clinical Centre, Ljubljana,

Table 2. Comparison of the performance of expert physicians and machine learning classification methods for the CAD dataset

	Positive			Negative		
Method	Reliable	Errors	AUC	Reliable	Errors	AUC
Physicians	0.72	0.04	0.790	0.45	0.07	0.650
RE_{SVM}	0.65	0.00	0.903	0.30	0.04	0.566
TRE_{SVM}	0.76	0.02	0.861	0.57	0.08	0.672
RE_{RVM}	0.63	0.004	0.842	0.54	0.06	0.729
TRE_{RVM}	0.67	0.013	0.767	0.49	0.05	0.702
$TRE_{RVM(inc)}$	0.69	0.004	0.850	0.54	0.07	0.720



(a) SVM, Positive (b) SVM, Negative

(c) RVM, Positive (d) RVM, Negative

Fig. 2. ROC curves for the transduction reliability measures for SVM and incremental RVM, using the CAD dataset and considering separately the positive and negative examples

Slovenia, between 1991 and 1994. We selected only the patients with complete diagnostic procedures (all four levels) [13].

Physicians apply a stepwise diagnostic process and use Bayes law to compute a posterior probability of disease, based on some diagnostic tests and a prior probability according to the age, gender and type of chest pain for each patient. Reliable diagnoses are assumed to be those whose posterior probability is over 0.90 (positive) or under 0.10 (negative). We considered treating the problem by training an SVM or an RVM classifier and using the transductive reliability principle to estimate the reliability of each classification. For evaluation purposes, we performed leave-one-out cross-validation, and for each example we predicted a class and a reliability of the classification. We then splitted classifications to reliable and unreliable by computing the threshold that maximizes the information gain and measured the percentage of reliable diagnoses (with the reliability measure above some threshold), and errors made in this process (percentage of incorrectly diagnosed patients with seemingly reliable diagnoses).

The results of these experiments are summariazed in Table 2 and furthermore, in Figure 2 ROC curves are plotted separately for the cases of positive and negative examples. The area (AUC) under these curves, which measures the

overall reliability performace, is also shown in Table 2. It can be observed that when the transduction principle is used along with SVM and incremental RVM, better performance is achieved compared to physicians.

Specifically, notice that the transductive SVM, has reliably detected 0.76% of the positive examples and 0.57% of negative examples, which is much better than the percentages of physicians, which are 0.72% and 0.45% respectively More important is the fact that at the same time, transductive SVM made less errors in positive examples, specifically 0.02% when the physicians made 0.04%. In medical diagnosis applications, it is very important that this error rate is kept at very small values. The error rate in negative examples, is 0.08% for physicians and 0.07% for transductive SVM, which is comparable.

When using the RVM model, the error rate of positive examples is dropped even lower to 0.004%. Although, the percentage of reliably detected positive examples (0.63%) is somewhat less than the one of physicians (0.72%), it is improved to 0.69% when using the incremental transduction principle. The RVM percentage of reliably detected negative examples is slightly higher than physicians, while the error rate of negative examples is about the same. Notice, that non-incremental transduction with RVM did not perform as expected, probably because relevant vectors are very sensitive to small changes of the training set.

6 Conclusions

We applied the transduction methodology for reliability estimation on sparse kernel-based classification methods. Experiments on medical datasets from the UCI repository and a bioinformatics gene expression dataset, indicate that, when used with kernel-based classifiers, transductive reliability estimations are more accurate than simple reliability measures based on the outputs of kernel classifiers. Furthermore, we applied the transductive methodology in the problem of CAD diagnosis, achieving better reliability estimation performance compared to the standard physicians procedure.

Acknowledgements

This work was supported in the framework of the "Bilateral S+T cooperation between the Hellenic Republic and the Republic of Slovenia (2004-2006)".

References

1. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press, Cambridge (2001)
2. Kukar, M., Kononenko, I.: Reliable classifications with Machine Learning. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS (LNAI), vol. 2430, Springer, Heidelberg (2002)
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory, pp. 92–100 (1998)

4. Li, M., Vitányi, P.: An introduction to Kolmogorov complexity and its applications, 2nd edn. Springer, New York (1997)
5. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Proc. ICML'95, pp. 194–202. Morgan Kaufmann, San Francisco (1995)
6. Gammerman, A., Vovk, V., Vapnik, V.: Learning by transduction. In: Proceedings of the 14^{th} Conference on Uncertainty in Artificial Intelligence, Madison, Wisconsin, pp. 148–155 (1998)
7. Saunders, C., Gammerman, A., Vovk, V.: Transduction with confidence and credibility. In: Proceedings of the International Joint Conference on Artificial Intelligence, Stockholm, Sweden (1999)
8. Proedrou, K., Nouretdinov, I., Vovk, V., Gammerman, A.: Transductive confidence machines for pattern recognition. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS (LNAI), vol. 2430, pp. 381–390. Springer, Heidelberg (2002)
9. Kukar, M.: Quality assessment of individual classifications in machine learning and data mining. Knowledge and Information Systems 9(3), 364–384 (2006)
10. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
11. Tipping, M.E.: Sparse Bayesian learning and the Relevance Vector Machine. Journal of Machine Learning Research 1, 211–244 (2001)
12. Tipping, M., Faul, A.: Fast marginal likelihood maximisation for sparse Bayesian models. In: Proc. of the Ninth International Workshop on Artificial Intelligence and Statistics (2003)
13. Kukar, M., Kononenko, I., selj, C.G., Kralj, K., Fettich, J.: Analysing and improving the diagnosis of ischaemic heart disease with machine learning. Artificial Intelligence in Medicine: Special Issue on Data Mining Techniques and Applications in Medicine (1999) (in press)

Parameter Learning for Bayesian Networks with Strict Qualitative Influences

Ad Feelders and Robert van Straalen

Utrecht University, Department of Information and Computing Sciences,
P.O. Box 80089, 3508TB Utrecht, The Netherlands
ad@cs.uu.nl, Robert.vanStraalen@phil.uu.nl

Abstract. We propose a new method for learning the parameters of a Bayesian network with qualitative influences. The proposed method aims to remove unwanted (context-specific) independencies that are created by the order-constrained maximum likelihood (OCML) estimator. This is achieved by averaging the OCML estimator with the fitted probabilities of a first-order logistic regression model. We show experimentally that the new learning algorithm does not perform worse than OCML, and resolves a large part of the independencies.

1 Introduction

In recent work, Wittig and Jameson [8], Altendorf et al. [1] and Feelders and van der Gaag [4] have shown that the use of qualitative influences can improve the probability estimates in Bayesian networks, in case relatively few observations are available. Apart from improvement of the parameter estimates, in [8] and [4] it was argued that networks with probability estimates that reflect the qualitative knowledge specified by the domain experts are less likely to exhibit counterintuitive reasoning behaviour and are therefore more likely to be accepted.

Feelders and van der Gaag [4] provide a simple algorithm, based on the isotonic regression, to compute the order-constrained maximum likelihood (OCML) estimates for networks of binary variables. A disadvantage of the OCML estimates is that in case order reversals are present in the unconstrained estimates, these are resolved by setting blocks of violating estimates equal to their weighted average. This results in unwanted (context-specific) independencies in the network.

We present a new estimator that aims at enforcing strict inequalities between parameters, thereby avoiding the creation of unwanted independencies in the network. This is achieved by combining the OCML with a first-order logistic regression model.

The paper is organized as follows. In section 2 we introduce the necessary notation, and introduce some important concepts that are used throughout the paper. In section 3 we discuss parameter learning with qualitative influences, and explain the shortcoming of the order-constrained maximum likelihood estimator.Subsequently, in section 4, we discuss our compound estimator that aims

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 48–58, 2007.

to remove this shortcoming. This new parameter learning method is evaluated experimentally in section 5. We end with conclusions.

2 Preliminaries

A *Bayesian network* is a concise representation of a joint probability distribution over a collection of stochastic variables $\mathbf{V} = (V_1, \ldots, V_m)$; in the sequel, we assume all variables to be binary, taking the value 0 or 1. The network consists of an acyclic directed graph in which each node corresponds to a variable and the arcs capture the dependence structure of the distribution; the network further includes a number of conditional probabilities, or *parameters*, $P(V_i \mid \mathbf{V}_{\mathrm{pa}(i)})$ for each variable V_i given its parents $\mathbf{V}_{\mathrm{pa}(i)}$ in the graph. The graphical structure and associated probabilities together represent a unique joint probability distribution over the variables involved, which is factorised according to

$$\Pr(\mathbf{V}) = \prod_{i=1}^{m} P(V_i \mid \mathbf{V}_{\mathrm{pa}(i)})$$

In estimating the parameters from data, we only have to consider one node and its parents at a time. To simplify notation, we will do so in the sequel. Let $\mathbf{X} = (X_1, \ldots, X_k)$ be the parents of a variable Y, and let $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_k = \{0, 1\}^k$ consist of vectors $\mathbf{x} = (x_1, x_2, \ldots, x_k)$ of values for the k variables in $\mathbf{X}$, that is, $\mathcal{X}$ is the set of all *parent configurations* of Y. Slightly abusing terminology, we sometimes say that X_i *occurs* or *is present* if it has the value one. We write $\mathbf{X}_a$ for the sub-vector of $\mathbf{X}$ containing the variables X_j for $j \in a$, where a is a subset of $K = \{1, \ldots, k\}$. We further write $\mathbf{X}_{-a}$ for $\mathbf{X}_{K \setminus a}$, and $\mathbf{X}_{-i}$ for $\mathbf{X}_{K \setminus \{i\}}$, where $i \in K$. Furthermore, we write for example $n(y, \mathbf{x})$ to denote the number of observations in the data with $Y = y$ and $\mathbf{X} = \mathbf{x}$.

A *qualitative influence* [7] between two variables expresses how observing a value for the one variable affects the probability distribution for the other variable. A *positive* influence of X_i on Y along an arc $X_i \rightarrow Y$ means that the occurrence of X_i *does not decrease* the probability that Y occurs, regardless of any other direct influences on Y, that is

$$P(y = 1 | x_i = 1, \mathbf{x}_{-i}) \geq P(y = 1 | x_i = 0, \mathbf{x}_{-i}), \tag{1}$$

where $\mathbf{x}_{-i}$ is any configuration of the parents of Y other than X_i. Since the inequality in (1) is not strict, the technically correct, if somewhat awkward, verbal description is *does not decrease* rather than *increases*. In this paper, the distinction between the two is crucial however. Similarly, there is a *negative* influence of X_i on Y along an arc $X_i \rightarrow Y$ if the occurrence of X_i *does not increase* the probability that Y occurs. From now on we assume, without loss of generality, that all qualitative influences are positive.

Finally, we say a positive qualitative influence is *strict* if the occurrence of X_i *increases* the probability that Y occurs, regardless of any other direct influences on Y, that is

$$P(y = 1 | x_i = 1, \mathbf{x}_{-i}) > P(y = 1 | x_i = 0, \mathbf{x}_{-i}). \tag{2}$$

3 Parameter Learning with Qualitative Influences

As we saw in the previous section, qualitative influences correspond to certain constraints between the parameters in the Conditional Probability Table (CPT) of Y. In earlier work, several methods have been proposed to exploit these constraints in estimating the parameters from data. In [4] it was shown how the order-constrained maximum likelihood (OCML) estimates, using the non-strict inequalities in (1), can be computed using the isotonic regression. The problem with these estimates is that they create unwanted (context-specific) independencies. This is illustrated by the following example.

Consider a node Y with two parents, X_1 and X_2, both with a positive influence on Y. Suppose we observe the data given in the left part of Table 1.

Table 1. Data for example. In the left table, each cell gives $n(y = 1, \mathbf{x})/n(\mathbf{x})$. The middle table gives the unconstrained ML estimates of $P(Y = 1|X_1, X_2)$. The table on the right gives the order-constrained ML estimates.

X_1/X_2	0	1
0	4/10	1/3
1	6/10	18/20

X_1/X_2	0	1
0	0.40	0.33
1	0.60	0.90

X_1/X_2	0	1
0	0.38	0.38
1	0.60	0.90

The first row of the unconstrained estimates contains an order reversal, since it is decreasing rather than increasing. The OCML estimator resolves this order violation by taking the weighted average of the two violating cells, and assigning this value to the both of them. Hence the value

$$\frac{4+1}{10+3} = \frac{5}{13} \approx 0.38$$

in the first row of the rightmost table in Table 1. The result is a context-specific independence: according to the OCML estimates, Y is independent of X_2 for $X_1 = 0$. This is probably not what the expert intended when she specified a positive influence of X_2 on Y. This raises the question if it wouldn't be better to enforce the strict inequalities given in equation (2) rather than the non-strict inequalities given in (1). The problem is that for the strict inequalities, the OCML estimates do not exist in case one of the order constraints is violated. This can be intuitively appreciated by looking at the solution obtained in the example above. In the strict version the likelihood would keep increasing as we make the difference between the violating estimates $\hat{P}(Y = 1|0, 0)$ and $\hat{P}(Y = 1|0, 1)$ smaller and smaller.

In order to enforce strict inequalities, Altendorf et al. [1] specified a minimum margin between two "contiguous" parameters, that is, (1) was replaced by

$$P(y = 1|x_i = 1, \mathbf{x}_{-i}) \geq P(y = 1|x_i = 0, \mathbf{x}_{-i}) + \varepsilon_{\mathbf{x}_{-i}}, \tag{3}$$

where $\varepsilon_{\mathbf{x}_{-i}}$ is the minimum difference required, which in general may depend on the values at which the remaining parents are held constant. The problem

with this approach is the selection of appropriate values for the $\varepsilon_{\mathbf{x}_{-i}}$. One could try to elicit these margins from domain experts. Experience shows however that experts have a hard time in reliably providing such numerical information. If they had been very good at it, we might just as well have asked them for the CPTs straight away. Another possibility is to specify some fixed value for the margin, that is applied to every pair of contiguous parameters. This option is chosen in [1]. The problem is, however, that the choice of value for ε becomes arbitrary.

Therefore, we propose a different, data-driven, approach to obtain strict inequalities between contiguous parameters. This approach is presented in the next section.

4 Learning with Strict Qualitative Influences

The basic idea of our approach is to try to remove unwanted independencies by combining the OCML estimates with estimates produced by a logistic regression model. A similar idea in the different setting of numeric isotonic regression with just one predictor variable was proposed by Wright [9]. For the logistic regression model, we assume the log-odds is linear in the parent variables, that is

$$\log\left\{\frac{P(Y=1|\mathbf{X})}{P(Y=0|\mathbf{X})}\right\} = \beta_0 + \sum_{i=1}^{k} \beta_i X_i \tag{4}$$

The important property of this model is that $\beta_i > 0$ corresponds to a strictly positive influence of X_i on Y. The proposed compound estimator $\hat{P}^*$ is given by

$$\hat{P}^*(Y|\mathbf{X}) = \gamma \hat{P}_{\text{LR}}(Y|\mathbf{X}) + (1-\gamma)\hat{P}_{\text{OCML}}(Y|\mathbf{X}),$$

where $0 < \gamma < 1$, and $\hat{P}_{\text{LR}}(Y|\mathbf{X})$ are the fitted probabilities of a first-order logistic regression model. By taking the weighted average with the fitted logistic regression probabilities, we enforce strict inequalities as long as the signs of the logistic regression coefficients are positive. There are still two loose ends in this proposal

1. What if one or more of the logistic regression coefficient estimates are negative instead of positive?
2. How do we choose the value of γ, the weight of the logistic regression model?

The first issue is addressed as follows. We start by estimating the full model, i.e. including all parents of Y as predictors. Then we check if there are any parents having a negative coefficient. If so, these parents are removed from the model, and the model is re-estimated with the remaining parents. If necessary, this procedure is repeated until only parents with positive coefficients remain. Note that the removal of X_i from the model (i.e. setting $\beta_i = 0$) results in

$$\hat{P}_{\text{LR}}(y=1|x_i=1,\mathbf{x}_{-i}) = \hat{P}_{\text{LR}}(y=1|x_i=0,\mathbf{x}_{-i}), \tag{5}$$

for every configuration $\mathbf{x}_{-i}$. Hence, if the OCML estimates satisfy

$$\hat{P}_{\text{OCML}}(y = 1|x_i = 1, \mathbf{x}_{-i}) = \hat{P}_{\text{OCML}}(y = 1|x_i = 0, \mathbf{x}_{-i}), \tag{6}$$

for one or more configurations $\mathbf{x}_{-i}$, these equalities will *not* be resolved by the compound estimator.

With respect to the second issue, we should keep in mind that the primary reason to combine the OCML estimates with the logistic regression model is to obtain data-driven margins between contiguous parameters. We do not want its weight to become too big, unless the LR model actually gives a good fit of the data. Therefore, we chose to let γ depend on the p-value of the observed deviance of the fitted logistic regression model, under the null hypothesis that the logistic regression model is the correct model specification.

To illustrate the basic idea of our compound estimator, we continue the example of section 3. Recall that the OCML estimates created an independence between X_2 and Y for $X_1 = 0$. To remove this independence, we combine the OCML estimates with those obtained by fitting a logistic regression model with Y as the response, and X_1 and X_2 as the predictors. This results in positive coefficients for both X_1 ($\hat{\beta}_1 = 1.47$) and X_2 ($\hat{\beta}_2 = 1.09$). Note that X_2 still has a positive coefficient, because the order reversal in the first row of the observed relative frequencies is more than compensated by the increasing second row. If the second row were also decreasing, X_2 would have had a negative coefficient, and consequently the problem could not be fixed by the compound estimator. The fitted probabilities of the logistic regression model are given in Table 2.

Table 2. Fitted probabilities of the logistic regression model estimated on the data in Table 1 on the left. On the right, the fitted probabilities of the compound estimator.

X_1/X_2	0	1
0	0.32	0.59
1	0.68	0.86

X_1/X_2	0	1
0	0.37	0.43
1	0.62	0.89

To compute the compound estimator, we still have to determine the weight of the LR model. Here we take γ equal to the p-value; in the experimental section, we also consider two other possibilities. We compute the p-value by using the fact that the deviance has approximately a $\chi^2_{n-k'-1}$ distribution under the null hypothesis, where k' is the number of remaining variables in the logistic regression model. Hence, we look up the area of the tail of this distribution to the right of the observed deviance, and find a p-value of approximately 0.26. Hence the compound estimator becomes

$$\hat{P}^*(Y = 1|X_1, X_2) = 0.26 \cdot \hat{P}_{\text{LR}}(Y = 1|X_1, X_2) + 0.74 \cdot \hat{P}_{\text{OCML}}(Y = 1|X_1, X_2),$$

The compound estimates are given in Table 2 on the right.We have achieved our goal: the unwanted equality in the first row has been resolved, and the margin between the two contiguous probabilities has been determined by the data.

5 Experiments

To evaluate the proposed parameter learning method, we performed experiments on both artificial data and real data. We are interested in two aspects of performance:

1. How good are the estimates produced by our compound estimator?
2. How many of the unwanted independencies in the OCML estimates are fixed, in the sense that our compound estimator turns them into strict inequalities?

Our objective is to remove as many unwanted independencies as possible, while retaining high quality estimates.

5.1 Artificial Data

In order to test our approach under various circumstances, we set up several environments to measure the performance of the proposed compound estimator. Parameters in these test environments are

1. The number of parent nodes: 2, 3 and 4.
2. The size of the data sample: 20, 50 and 100 for 2 or 3 parents; 50, 100 and 200 for 4 parents.
3. The margins between the conditional probabilities $P(Y = 1|\mathbf{X})$ for contiguous values of $\mathbf{X}$ of the underlying true distribution: small and large.

To elaborate on the third point: we suspect that with small margins, reversed signs of the coefficients in the LR model will occur more often, leading to a lower fraction of resolved independencies. An example of small and large margins for the case of two parents is given in Figure 1.

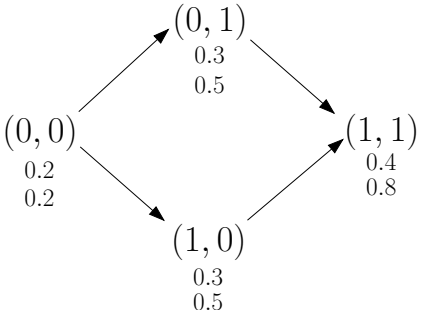


Fig. 1. Small and large margins between $P(Y = 1|\mathbf{X})$ for contiguous values of $\mathbf{X}$. The first number below each parent configuration $\mathbf{x}$ denotes $P(Y = 1|\mathbf{x})$ corresponding to the small margin; the second number corresponds to the large margin. Arrows between the parent configurations denote direct precedence in the order ("contiguous" configurations).

We considered three definitions of the LR weight γ in the experiments:

1. γ_1 = p-value: Just the p-value as discussed in section 4.
2. $\gamma_2 = \min(1, \frac{2^{|\mathcal{X}|}}{n} \cdot \text{p-value})$.
3. $\gamma_3 = \min(0.1, 10 \cdot \text{p-value})$: The p-value is multiplied by 10, to avoid γ being too small. Also it has an upper threshold of 0.1 to avoid too large values. Here an extra factor is included, which decreases as the sample size n (relative to the number of parameters) increases. It has a maximum of 1 and goes to 0 as n goes to infinity. This definition is inspired by the view that with more data, the ML (and OCML) estimates become more reliable, requiring a smaller weight for the LR model.

It is clear that none of these three weighting methods has any deep theoretical justification, so the experiments will have to make clear which one works best.

In our experiments we fitted five different models: $\hat{P}_{ML}$ (the unconstrained maximum likelihood estimates), $\hat{P}_{OCML}$ (the order-constrained maximum likelihood estimates), and $\hat{P}_i^*$, $i = 1, 2, 3$ (the compound estimates, using γ_i). To determine the quality of the estimates, we computed the Kullback-Leibler (KL) divergence between the true and the fitted distributions. We applied the Laplace correction to avoid possible infinity values. We performed a 1000 replications of the experiment, and averaged the Kullback-Leibler divergence over these 1000 replications. The results are given in Tables 3, 4, and 5, for 2,3, and 4 parents respectively.

Table 3. Results for artificial data, 2 parent nodes

ntrain	KL_{ML}	KL_{OCML}	KL_{C1}	KL_{C2}	KL_{C3}	fixratio
distribution with large margins						
20	0.0598	0.0436	0.0692	0.0448	0.0443	63%
50	0.0319	0.0277	0.0278	0.0277	0.0274	89%
100	0.0185	0.0172	0.017	0.0172	0.017	99%
distribution with small margins						
20	0.0576	0.0342	0.0539	0.0353	0.0345	31%
50	0.0305	0.0197	0.0207	0.0197	0.0195	43%
100	0.0184	0.0129	0.013	0.0129	0.0128	49%

There are several general observations clearly shown in the tables. First of all, the ML model has the worst performance. Second, the compound estimator using γ_1 does not perform well: the LR model becomes too dominant and spoils the estimates. Third, the OCML model and the models using γ_2 and γ_3 are close, but rather consistently the compound estimator using γ_3 performs the best by a small margin. Finally, as the sample size increases, the differences in performance become smaller. Summarizing, we can say that with the right weight (γ_3) for the LR model, the compound estimator works well under all circumstances we have studied in the experiments.

Table 4. Results for artificial data, 3 parent nodes

ntrain	KL_{ML}	KL_{OCML}	KL_{C1}	KL_{C2}	KL_{C3}	fixratio
distribution with large margins						
20	0.0718	0.0325	0.0719	0.0381	0.0339	58%
50	0.0514	0.0286	0.0299	0.0287	0.0282	78%
100	0.0329	0.0223	0.0219	0.0223	0.0212	91%
distribution with small margins						
20	0.0702	0.0324	0.0868	0.037	0.0332	42%
50	0.0519	0.0231	0.025	0.0233	0.0228	48%
100	0.0325	0.0163	0.0163	0.0163	0.0157	62%

Table 5. Results for artificial data, 4 parent nodes

ntrain	KL_{ML}	KL_{OCML}	KL_{C1}	KL_{C2}	KL_{C3}	fixratio
distribution with large margins						
50	0.0716	0.0314	0.0348	0.0313	0.0302	84%
100	0.0522	0.0265	0.0256	0.0263	0.0247	93%
200	0.0336	0.0213	0.0206	0.0212	0.0196	99%
distribution with small margins						
50	0.0732	0.0312	0.038	0.0304	0.0289	59%
100	0.0516	0.0215	0.0212	0.0213	0.02	71%
200	0.0321	0.0143	0.0138	0.0143	0.0132	85%

The next question is, what fraction of the unwanted independencies created by OCML are resolved by the compound estimator. This fraction is referred to in the following as the *fixratio*. As we expected, the fixratio is higher for large margins, and it also increases with the sample size. Table 6 shows the average fixratios.

Table 6. Average fixratios

distribution	average fixratio
large margins	84%
small margins	54%
overall	69%

5.2 Real Data

We also performed experiments on five real datasets, four of which have been taken from the UCI repository [2]. All non-binary variables were made binary by making approximately equal-frequency bins of 0 and 1-values. We used the dependent variable and a selection from the attributes to construct a small Bayesian network fragment. To determine the sign of the influences, we used the

Table 7. Average log-loss for respectively the Windsor housing, Wisconsin breast cancer, Pima Indians, Haberman survival, and CPU performance datasets for different sizes of the training sample

ntrain	$\mathcal{L}_{ML}$	$\mathcal{L}_{OCML}$	$\mathcal{L}_{C1}$	$\mathcal{L}_{C2}$	$\mathcal{L}_{C3}$	fixratio
20	0.5888	0.5667	0.7624	0.5689	0.5645	70%
	(± 0.043)	(± 0.026)	(± 0.859)	(± 0.032)	(± 0.027)	
50	0.5539	0.5428	0.5634	0.5427	0.5418	89%
	(± 0.027)	(± 0.02)	(± 0.123)	(± 0.02)	(± 0.02)	
100	0.5354	0.5293	0.5305	0.5291	0.5284	99%
	(± 0.018)	(± 0.015)	(± 0.022)	(± 0.015)	(± 0.015)	
20	0.2999	0.2944	1.1285	0.2677	0.2853	5%
	(± 0.03)	(± 0.023)	(± 1.13)	(± 0.031)	(± 0.024)	
50	0.2415	0.2396	0.4461	0.2357	0.237	82%
	(± 0.017)	(± 0.015)	(± 0.435)	(± 0.015)	(± 0.015)	
100	0.2189	0.2182	0.2458	0.2175	0.2174	91%
	(± 0.011)	(± 0.011)	(± 0.156)	(± 0.01)	(± 0.01)	
20	0.6526	0.6164	0.6901	0.6198	0.6151	54%
	(± 0.042)	(± 0.024)	(± 0.332)	(± 0.03)	(± 0.025)	
50	0.6223	0.6024	0.6085	0.6023	0.6011	72%
	(± 0.027)	(± 0.018)	(± 0.067)	(± 0.018)	(± 0.018)	
100	0.6008	0.5916	0.5909	0.5914	0.5904	79%
	(± 0.018)	(± 0.013)	(± 0.014)	(± 0.013)	(± 0.013)	
20	0.6157	0.5872	0.7017	0.5883	0.5843	41%
	(± 0.043)	(± 0.028)	(± 0.572)	(± 0.034)	(± 0.03)	
50	0.5837	0.5648	0.5735	0.5647	0.5635	47%
	(± 0.027)	(± 0.019)	(± 0.057)	(± 0.02)	(± 0.02)	
100	0.5657	0.5539	0.5547	0.5538	0.5531	56%
	(± 0.026)	(± 0.021)	(± 0.026)	(± 0.021)	(± 0.021)	
20	0.499	0.4883	0.7738	0.4877	0.4849	56%
	(± 0.036)	(± 0.025)	(± 0.9)	(± 0.032)	(± 0.026)	
50	0.4728	0.4673	0.5248	0.4665	0.466	84%
	(± 0.028)	(± 0.024)	(± 0.249)	(± 0.026)	(± 0.025)	
100	0.4605	0.4579	0.4603	0.4572	0.4568	97%
	(± 0.038)	(± 0.038)	(± 0.061)	(± 0.038)	(± 0.038)	

information attached to the datasets, previous articles [1] and [3], and commonsense. We did not look at the data itself to determine the signs. The data sets used are:

- *Windsor housing data*: used as an example by Koop [5]. It has 546 examples of sale prices of houses dependent on different variables. We take the sale price as the child variable, and lot size and presence or absence of a basement and air conditioning as parent nodes with positive influences.
- *Wisconsin Breast Cancer Database* [6]: 699 instances of data about the class (benign or malignant) of breast tumors. We used the class as the child variable and clump thickness, single epithelial cell size and bland chromatin as parent nodes, all with positive influences.

Table 8. Average fixratios per dataset

dataset	average fixratio
Windsor housing	86%
Wisconsin breast cancer	60%
Pima indians diabetes	68%
Haberman survival	48%
CPU performance	79%
Overall	68%

Table 9. Average fixratios per sample size

sample size	average fixratio
20	45%
50	75%
100	84%
Overall	68%

- *Pima Indians diabetes*: 768 cases of medical data on Pima Indians. The class variable denotes whether the person has diabetes or not. We used the body mass index, diabetes pedigree function and age of the person as parent nodes, all with positive influences.
- *Haberman's Survival Data*: 306 cases from a study on the survival of patients who had undergone surgery for breast cancer. Survival of the patient is the child variable. The patient's age, year of operation and number of nodes detected are the parent nodes, again all with positive influences.
- *CPU Performance Data*: 209 cases of characteristics and performance of various types of computer CPU models. Performance is the dependent variable. Machine cycle time, minimum main memory and maximum main memory are the parent nodes. Machine cycle time has a negative influence, the other two parents have a positive influence.

Like with the artificial data, we performed a thousand replications of the experiment. Table 7 shows the results. For all estimators, we computed the average log-loss and its standard error on the data not used for training as follows:

$$\mathcal{L}_{\text{test}} = \frac{-\sum_{i=1}^{\text{ntest}} \log \hat{P}(y_i|\boldsymbol{x}_i)}{\text{ntest}},$$

where ntest is the number of observations in the test set. The results are fairly in line with those obtained on the artificial data. Model C3 (the compound model using γ_3) seems to perform slightly better than the other estimators again. Table 8 shows the average fixratios per dataset. Table 9 shows the average fixratios per sample size. The overall fixratio of the compound estimator is about 68%, and increases with the sample size.

6 Conclusions

We have proposed a new method for learning the parameters of a Bayesian network with qualitative influences. This method aims at avoiding unwanted independencies created by the order-constrained maximum likelihood (OCML) estimator. To obtain data-driven margins between contiguous parameters, the OCML estimates are combined with the fitted probabilities of a first-order logistic regression model. We have shown that the new compound estimator performs well, and is able to remove a large fraction of the independencies created by the OCML estimator.

References

1. Altendorf, E.A., Restificar, A.C., Dietterich, T.G.: Learning from sparse data by exploiting monotonicity constraints. In: Bacchus, F., Jaakkola, T. (eds.) Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI-05), pp. 18–25. AUAI Press (2005)
2. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998), http://www.ics.uci.edu/~mlearn/mlrepository.html
3. Feelders, A., Pardoel, M.: Pruning for monotone classification trees. In: Berthold, M.R., Lenz, H.-J., Bradley, E., Kruse, R., Borgelt, C. (eds.) IDA 2003. LNCS, vol. 2810, pp. 1–12. Springer, Heidelberg (2003)
4. Feelders, A., van der Gaag, L.: Learning Bayesian network parameters with prior knowledge about context-specific qualitative influences. In: Bacchus, F., Jaakkola, T. (eds.) Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI-05), pp. 193–200. AUAI Press (2005)
5. Koop, G.: Analysis of Economic Data. John Wiley and Sons, Chichester (2000)
6. Mangasarian, O.L., Nick Street, W., Wolberg, W.H.: Breast cancer diagnosis and prognosis via linear programming. Technical Report MP-TR-1994-10 (1994)
7. Wellman, M.P.: Fundamental concepts of qualitative probabilistic networks. Artificial Intelligence 44, 257–303 (1990)
8. Wittig, F., Jameson, A.: Exploiting qualitative knowledge in the learning of conditional probabilities of Bayesian networks. In: Boutilier, C., Goldszmidt, M. (eds.) Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, pp. 644–652. Morgan Kaufmann, San Francisco (2000)
9. Wright, F.T.: Estimating strictly increasing regression functions. Journal of the American Statistical Association 73(363), 636–639 (1978)

Tree Augmented Naive Bayes for Regression Using Mixtures of Truncated Exponentials: Application to Higher Education Management*

Antonio Fernández, María Morales, and Antonio Salmerón

Department of Statistics and Applied Mathematics, University of Almería,
Carrera de Sacramento s/n, E-04120 Almería, Spain
{afalvarez,maria.morales,antonio.salmeron}@ual.es

Abstract. In this paper we explore the use of Tree Augmented Naive Bayes (TAN) in regression problems where some of the independent variables are continuous and some others are discrete. The proposed solution is based on the approximation of the joint distribution by a Mixture of Truncated Exponentials (MTE). The construction of the TAN structure requires the use of the conditional mutual information, which cannot be analytically obtained for MTEs. In order to solve this problem, we introduce an unbiased estimator of the conditional mutual information, based on Monte Carlo estimation. We test the performance of the proposed model in a real life context, related to higher education management, where regression problems with discrete and continuous variables are common.

1 Introduction

In real life applications, it is common to find problems in which the goal is to predict the value of a variable of interest depending on the values of some other observable variables. If the variable of interest is discrete, we are faced with a *classification problem*, whilst if it is continuous, it is usually called a *regression problem*. In classification problems, the variable of interest is called *class* and the observable variables are called *features*, while in regression frameworks, the variable of interest is called *dependent variable* and the observable ones are called *independent variables*.

Bayesian networks [8,12] have been previously used for classification and regression purposes. More precisely, naive Bayes models have been applied to regression problems under the assumption that the joint distribution of the independent variables and the dependent variable is multivariate Gaussian [7]. If the normality assumption is not fulfilled, the problem of regression with naive Bayes models has been approached using kernel densities to model the conditional distribution in the Bayesian network [5], but the obtained results are poor. Furthermore, the use of

* This work has been supported by the Spanish Ministry of Education and Science, project TIN2004-06204-C03-01 and by Junta de Andalucía, project P05-TIC-00276.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 59–69, 2007.

kernels introduce a high complexity in the model, which can be problematic, especially because standard algorithms for carrying out the computations in Bayesian networks are not valid for kernels. A restriction of Gaussian models is that they only apply to scenarios in which all variables are continuous.

Motivated by the application to higher education management, we are interested in regression problems where some of the independent variables are discrete while the others are continuous. Therefore, the joint distribution is not multivariate Gaussian in any case, due to the presence of discrete variables. Recently, a naive Bayes regression model based on the approximation of the joint distribution by a Mixture of Truncated Exponentials (MTE) was proposed [11]. The MTE model has been successfully used in the context of Bayesian networks as a solution to the presence of discrete and continuous variables simultaneously [9], showing good features as an exact model as well as an approximation of other probability distributions [2,3].

In this paper we study the extension of the naive Bayes regression model based on MTEs [11] to tree augmented naive Bayes (TAN) structures, in which the existence of dependencies between the *independent* variables is allowed. In order not to use a misleading terminology, from now on we will refer to the observable variables as *features*, even if we are in a regression context.

The rest of the paper is organised as follows. The use of Bayesian networks for regression is explained in section 2. The MTE model is described in section 3. Section 4 is devoted to our proposal of a TAN model for regression and its application to prediction in higher education is contained in section 5. Further experimental tests are described in section 6. The paper ends with conclusions in section 7.

2 Bayesian Networks and Regression

Throughout this paper, random variables will be denoted by capital letters, and their values by lowercase letters. In the multi-dimensional case, boldfaced characters will be used. The domain of the variable $\mathbf{X}$ is denoted by $\Omega_{\mathbf{X}}$. A *Bayesian network* for a set of variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a directed acyclic graph where each variable is assigned to one node, which has associated a conditional distribution given its parents [8,12]. An arc linking two variables indicates the existence of probabilistic dependence between both of them. An important feature of Bayesian networks is that the joint distribution over $\mathbf{X}$ factorises according to the d-separation criterion as follows [12]:

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

where $Pa(X_i)$ denotes the set of parents of variable X_i and $pa(x_i)$ is a configuration of values of them.

A Bayesian network can be used as a regression model for prediction purposes. Assume that Y is the dependent variable and $X_1, \ldots, X_n$ are the features. Then,

in order to predict the value for Y for observed features $x_1, \ldots, x_n$ the conditional density

$$f(y|x_1, \ldots, x_n) \ , \tag{2}$$

is computed and a numerical prediction for Y is given using the corresponding mean, the median or the mode.

As $f(y|x_1, \ldots, x_n)$ is proportional to $f(y) \times f(x_1, \ldots, x_n|y)$, solving the regression problem requires to specify an n dimensional density for $X_1, \ldots, X_n$ given Y. Using the factorisation determined by the Bayesian network, this problem is simplified depending on the structure of the network. The extreme case is the so-called *naive Bayes* structure [4,6], where all the feature variables are considered independent given the dependent variable. This kind of structure is represented in figure 1. This is the model used for regression in [11].

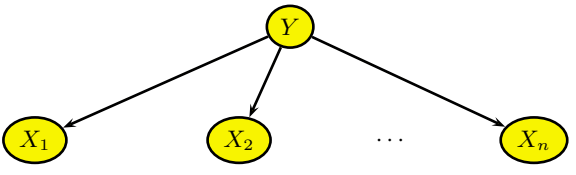


Fig. 1. Naive Bayes structure for regression. The observable variables are assumed to be independent given Y.

The reason to make the strong independence assumption behind naive Bayes models is that it is compensated by the reduction of the number of parameters to be estimated from data, since in this case, it holds that

$$f(y|x_1, \ldots, x_n) = f(y) \prod_{i=1}^{n} f(x_i|y) \ , \tag{3}$$

which means that, instead of one n-dimensional conditional densities, n one-dimensional conditional densities are estimated.

A compromise between the strong independence assumption and the complexity of the model to be estimated from data is the so-called *tree augmented naive Bayes* (TAN) [6]. In this kind of models, some more dependencies are allowed, expanding the naive Bayes structure by permitting each feature to have one more parent besides Y. In other words, the features form a tree (see figure 2).

In any case, regardless of the structure employed, it is necessary that the joint distribution for $Y, X_1, \ldots, X_n$ follows a model that allows the computation of the density in equation (2). Furthermore, taking into account our application purposes, it should be a model that allows the simultaneous presence of discrete and continuous variables. We think that the model that best fulfills these requirements is the MTE model, explained next.

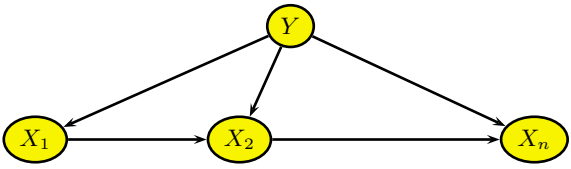


Fig. 2. A TAN structure for regression. Some more dependencies between the features are allowed.

3 The MTE Model

The MTE model can be defined as follows [9]:

Definition 1. (MTE potential) *Let* $\mathbf{X}$ *be a mixed* n*-dimensional random vector. Let* $\mathbf{Y} = (Y_1, \ldots, Y_d)$ *and* $\mathbf{Z} = (Z_1, \ldots, Z_c)$ *be the discrete and continuous parts of* $\mathbf{X}$*, respectively, with* $c + d = n$*. We say that a function* $f : \Omega_{\mathbf{X}} \mapsto \mathbb{R}_0^+$ *is a* Mixture of Truncated Exponentials potential (MTE potential) *if one of the next conditions holds:*

i. $\mathbf{Y} = \emptyset$ *and* f *can be written as*

$$f(\mathbf{x}) = f(\mathbf{z}) = a_0 + \sum_{i=1}^{m} a_i \exp\left\{\sum_{j=1}^{c} b_i^{(j)} z_j\right\} \tag{4}$$

for all $\mathbf{z} \in \Omega_{\mathbf{Z}}$*, where* a_i*,* $i = 0, \ldots, m$ *and* $b_i^{(j)}$*,* $i = 1, \ldots, m$*,* $j = 1, \ldots, c$ *are real numbers.*

ii. $\mathbf{Y} = \emptyset$ *and there is a partition* $D_1, \ldots, D_k$ *of* $\Omega_{\mathbf{Z}}$ *into hypercubes such that* f *is defined as*

$$f(\mathbf{x}) = f(\mathbf{z}) = f_i(\mathbf{z}) \quad \text{if} \quad \mathbf{z} \in D_i \ ,$$

where each f_i*,* $i = 1, \ldots, k$ *can be written in the form of equation (4).*

iii. $\mathbf{Y} \neq \emptyset$ *and for each fixed value* $\mathbf{y} \in \Omega_{\mathbf{Y}}$*,* $f_{\mathbf{y}}(\mathbf{z}) = f(\mathbf{y}, \mathbf{z})$ *can be defined as in* ii.

Example 1. The function f defined as

$$f(z_1, z_2) = \begin{cases} 2 + e^{3z_1+z_2} + e^{z_1+z_2} & \text{if} \ \ 0 < z_1 \leq 1, \ 0 < z_2 < 2 \\ 1 + e^{z_1+z_2} & \text{if} \ \ 0 < z_1 \leq 1, \ 2 \leq z_2 < 3 \\ \dfrac{1}{4} + e^{2z_1+z_2} & \text{if} \ \ 1 < z_1 < 2, \ 0 < z_2 < 2 \\ \dfrac{1}{2} + 5e^{z_1+2z_2} & \text{if} \ \ 1 < z_1 < 2, \ 2 \leq z_2 < 3 \end{cases}$$

is an MTE potential since all of its parts are MTE potentials.

Definition 2. (MTE density) *An MTE potential f is an* MTE density *if*

$$\sum_{\mathbf{y}\in\Omega_{\mathbf{Y}}}\int_{\Omega_{\mathbf{Z}}} f(\mathbf{y},\mathbf{z})d\mathbf{z} = 1 \ .$$

A *conditional MTE density* can be specified by dividing the domain of the conditioning variables and specifying an MTE density for the conditioned variable for each configuration of splits of the conditioning variables. Moral et al. [9] propose a data structure to represent MTE potentials, which is specially appropriate for this kind of conditional densities: The so-called *mixed probability trees* or mixed trees for short. The formal definition is as follows:

Definition 3. (Mixed tree) *We say that a tree $\mathcal{T}$ is a* mixed tree *if it meets the following conditions:*

i. *Every internal node represents a random variable (discrete or continuous).*
ii. *Every arc outgoing from a continuous variable Z is labeled with an interval of values of Z, so that the domain of Z is the union of the intervals corresponding to the arcs emanating from Z.*
iii. *Every discrete variable has a number of outgoing arcs equal to its number of states.*
iv. *Each leaf node contains an MTE potential defined on variables in the path from the root to that leaf.*

Mixed trees can represent MTE potentials defined by parts. Each entire branch in the tree determines one sub-region of the space where the potential is defined, and the function stored in the leaf of a branch is the definition of the potential in the corresponding sub-region.

4 Constructing a TAN Using MTEs

The construction of a TAN from a database can be decomposed into two tasks:

1. Determining the structure of the network (i.e., finding the dependencies between the features).
2. Estimating the MTE densities corresponding to the obtained structure.

Regarding the second task, existing algorithms for estimating the parameters of an MTE density are based on least squares estimation [15,17]. The construction of conditional densities is studied in [10], following ideas used for the construction of classification trees [14]. In this work we use the algorithm proposed in [10] for learning the conditional distributions, with the parameter learning approach described in [15,17].

With respect to obtaining the dependence structure among the features, the goal is to find a tree structure containing them [6], so that the links of the tree connect the variables with higher degree of dependence. This task can be solved

using a variation of the method proposed in [1]. The idea is to start with a fully connected graph, labeling the links with the conditional mutual information between the connected features given the independent variable (Y). Afterwards, the tree structure is obtained by computing the maximum spanning tree of the initial labeled graph. The detailed algorithm could be as follows:

Algorithm TAN_mte
INPUT: A database D with variables $Y, X_1, \ldots, X_n$.
OUTPUT: A TAN with root variable Y and features $X_1, \ldots, X_n$, with joint distribution of class MTE.

1. Construct a complete graph $\mathcal{C}$ with nodes $X_1, \ldots, X_n$.
2. Label each link (X_i, X_j) with the conditional mutual information between X_i and X_j given Y, i.e.,

$$I(X_i, X_j|Y) = \iiint f(x_i, x_j, y) \log \frac{f(x_i, x_j|y)}{f(x_i|y)f(x_j|y)} dx_i dx_j dy \ . \qquad (5)$$

3. Let $\mathcal{T}$ be the maximum spanning tree obtained from $\mathcal{C}$.
4. Direct the links in $\mathcal{T}$ in such a way that no node has more than one parent.
5. Construct a new network $\mathcal{G}$ with nodes $Y, X_1, \ldots, X_n$ and the same links as $\mathcal{T}$.
6. Insert the links $Y \rightarrow X_i$, $i = 1, \ldots, n$ in $\mathcal{G}$.
7. Estimate an MTE density for Y, and a conditional MTE density for each X_i, $i = 1, \ldots, n$ given its parents in $\mathcal{G}$.
8. Let P be the set of estimated densities.
9. Let TAN be a Bayesian network with structure $\mathcal{G}$ and distributions P.
10. Return TAN.

The main problem for implementing the algorithm above is the computation of the conditional mutual information defined in equation (5). The problem has been addressed for the Conditional Gaussian model [13], but only in classification contexts, i.e., with variable Y being discrete. For MTEs, the integral in equation (5) cannot be computed in closed form. Therefore, we propose to estimate it in a similar way as in [11] for the marginal mutual information. Our proposal is based on the estimator given in the next proposition.

Proposition 1. *Let X_i, X_j and Y be continuous random variables with joint MTE density $f(x_i, x_j, y)$. Let $(X_i^{(1)}, X_j^{(1)}, Y^{(1)}), \ldots, (X_i^{(m)}, X_j^{(m)}, Y^{(m)})$ be a sample of size m drawn from $f(x_i, x_j, y)$. Then,*

$$\hat{I}(X_i, X_j|Y) = \frac{1}{m} \sum_{k=1}^{m} \left(\log f(X_i^{(k)}|X_j^{(k)}, Y^{(k)}) - \log f(X_i^{(k)}|Y^{(k)}) \right) \qquad (6)$$

is an unbiased estimator of $I(X_i, X_j|Y)$.

Proof

$$
\begin{aligned}
E[\hat{I}(X_i, X_j|Y)] &= E\left[\frac{1}{m}\sum_{k=1}^{m}\left(\log f(X_i^{(k)}|X_j^{(k)}, Y^{(k)}) - \log f(X_i^{(k)}|Y^{(k)})\right)\right] \\
&= E[\log f(X_i|X_j, Y)] - E[\log f(X_i|Y)] \\
&= E[\log f(X_i|X_j, Y) - \log f(X_i|Y)] = E\left[\log \frac{f(X_i|X_j, Y)}{f(X_i|Y)}\right] \\
&= \iiint f(x_i, x_j, y) \log \frac{f(X_i|X_j, Y)}{f(X_i|Y)} dx_i dx_j dy \\
&= \iiint f(x_i, x_j, y) \log \frac{f(X_i|X_j, Y) f(X_j|Y)}{f(X_i|Y) f(X_j|Y)} dx_i dx_j dy \\
&= \iiint f(x_i, x_j, y) \log \frac{f(X_i, X_j|Y)}{f(X_i|Y) f(X_j|Y)} dx_i dx_j dy \\
&= I(X_i, X_j|Y) \ .
\end{aligned}
$$

Proposition 1 can be extended for the case in which X_i or X_j are discrete by replacing the corresponding integral by summation.

Therefore, the procedure for estimating the conditional mutual information consists of getting a sample from $f(x_i, x_j, y)$ and evaluating expression (6). Sampling from an MTE density is described in [16].

5 Application to Prediction in Higher Education Management

We have tested the performance of the TAN regression model based on MTEs in three practical problems related to higher education management, described in [11]. Our goal is to compare the accuracy of the TAN model versus the naive Bayes model used in [11].

Problems 1 and 2 consist of predicting the *performance rate* and *success rate*, respectively, for a given degree program. The study is restricted to a database with information about all the degree programs in the university of Almería in years 2001 to 2004.

The *performance rate* is defined as

$$pr = \frac{n_o}{n_s} \tag{7}$$

where n_s is the number of credits (blocks of 10 hours of confrontation lectures) of all the subjects selected by the students in a given year, and n_o is the number of credits actually obtained by the students at the end of the same year.

The *success rate* is defined as

$$sr = \frac{n_o}{n_e} \tag{8}$$

where n_o is as defined above and n_e is the number of credits for which the students actually took the final exam.

In problems 1 and 2, we have the following feature variables:

- **Degree:** The degree program.
- **OptRate:** Rate of credits that can be obtained with subjects freely chosen by the student.
- **OptOffer:** Number of free-election subjects offered (in credits) divided by the number of free-election credits that the student has to obtain to complete the degree.
- **Prt:** Ratio between the amount of practical credits and total amount of credits required in a degree program.
- **SmallGroups:** Ratio between the number of classes with no more than 20 students and the global number of classes in each subject.
- **BigGroups:** Ratio between the number of classes with not less than 80 students and the global number of classes in each subject.
- **Dedication:** Number of credits coursed by the students divided by the number of students.
- **Give-upRate:** Rate of students that leave the university without having finished the degree program.
- **Rate_S-L:** Number of students per lecturer.
- **PhD:** Fraction of credits in the degree given by lecturers owning the PhD degree.

Problem 3 consists of predicting the number of students in a given subject in year 2005. In this case, we have a database containing information about all the subjects offered at the University of Almería from years 2001 to 2004. The variables considered in this case are:

- **Degree:** The degree program.
- **Period:** Part of the degree (first or second half) in which the subject is located.
- **Subject.**
- **Course:** The course (year) in which the subject is located.
- **AXX:** Number of students in the given subject in year XX, ranging from 01 to 04.
- **prXX:** Performance rate for the given subject in year XX, ranging from 01 to 04.

Notice that in problems 1 and 2 the variable of interest is continuous, and in problem 3 it is discrete, but given its high number of possible values, its probability function cannot be handled appropriately using a probability table. Regarding the features, some of them are continuous while some others are discrete or even qualitative.

In the experiments reported in this section, we divided the domain of the continuous variables into 4 intervals, and all the necessary densities are represented using the so-called 5-parameter MTE:

$$f(x) = a_0 + a_1 e^{a_2 x} + a_3 e^{a_4 x}, \quad \alpha < x < \beta \ . \tag{9}$$

The reason to use the 5-parameter MTE is that it has shown its ability to fit the most common distributions accurately, while the model complexity and the number of parameters to estimate is low [3].

The accuracy of the models is measured using the mean squared error between the actual value y and its prediction $\hat{y}$, which is computed from a test set different to the one used for training the model:

$$rmse = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2} \ . \tag{10}$$

The predictions have been carried out using the mean and the median of the posterior distributions for Y. The results are reported in tables 1 and 2, where NB(mean) and NB(median) indicate the naive Bayes model predicting with the mean and the median respectively, and TAN(mean) and TAN(median) refer to the TAN model. The errors in table 1 have been computed dividing the database in two parts, one of them containing the 70% of the records in the original database, selected at random, which is used for training the model, and the other containing the remaining 30%, which is used for testing the model. The errors reported in table 2 have been computed using 10-fold cross validation.

Table 1. Results for the case study in terms of *rmse* measured in a hold-out set

	Problem 1	Problem 2	Problem 3
NB(mean)	0.1378	0.0679	33.3896
NB(median)	0.1458	0.0714	34.2925
TAN(mean)	**0.1324**	**0.0597**	**29.3093**
TAN(median)	0.1349	0.0609	29.9508

Table 2. Results for the case study in terms of *rmse* computed by 10-fold cross validation

	Problem 1	Problem 2	Problem 3
NB(mean)	0.0981	0.0441	36.9315
NB(median)	0.0989	0.0449	37.3671
TAN(mean)	**0.0913**	**0.0418**	**35.5599**
TAN(median)	0.0967	0.0452	36.7030

The results obtained in this analysis support the validity of the TAN model based on MTE for this case study. In the three problems considered, the TAN provides the best accuracy, especially in the case of predicting the number of students in a subject. This quantity is very difficult to predict, since the possible number of students in a subject varies from 0 to close to 500.

6 Further Experimental Evaluation

In the experiments reported in section 5 we found out that the way in which the links between the features are directed in step 4 of algorithm TAN_mte, influences the accuracy of the model. In fact, the results reported in tables 1 and 2 refer to

the best TAN models found, but for some orientations of the links it is possible to obtain TAN models that provide errors slightly higher than the naive Bayes. The difference in accuracy can be explained taking into account that the order of the variables in a mixed tree representing a conditional distribution determines the partition of the database which is used to estimate each one of the leaves in the tree. Therefore, small differences may appear if the data is scarce. In order to test this hypothesis, we carried out one more experiment using a larger database with more variables. More precisely, we used the `bodyfat` database from StatLib (`www.statlib.org`), which consists of 252 records containing data about 15 continuous variables. The results obtained measuring the error using 10-fold cross validation are reported in table 3. The errors for the TAN models correspond to the best model found. In this case, the differences between the different link orientations are only slight and in any case, the accuracy of the tan is always superior to the accuracy of the naive Bayes.

Table 3. Results for the `bodyfat` database computed by 10-fold cross validation

	NB(mean)	NB(median)	TAN(mean)	TAN(median)
rmse	11.9997	12.2897	**10.7441**	11.2067

7 Conclusions

We have introduced a TAN model for regression based on the use of MTEs. The aim was to approach a problem in which existing methods are difficult to apply due to the presence of discrete and continuous variables simultaneously. The obtained results are promising, as the TAN is competitive in the three problems considered. Also in the test case considered outside the scope of our main case study, the performance of the TAN model improves the naive Bayes.

In the future we expect to carry out a deeper analysis of the performance of our TAN model in other problems. Also, we think that it can be improved by incorporating a variable selection scheme. As we pointed out in section 6, the way in which the variables are ordered when directing the maximum spanning tree associated with the TAN can affect the accuracy of the final model, since the set of conditional densities to estimate may vary. This fact can be very important, especially in problems where the database is small, since the available data for estimating some conditionals can be scarce. Therefore, we think that a method for selecting an optimal order should be sought.

References

1. Chow, C.K., Liu, C.N.: Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory 14, 462–467 (1968)
2. Cobb, B., Rumí, R., Salmerón, A.: Modeling conditional distributions of continuous variables in Bayesian networks. In: Famili, A.F., Kok, J.N., Peña, J.M., Siebes, A., Feelders, A. (eds.) IDA 2005. LNCS, vol. 3646, pp. 36–45. Springer, Heidelberg (2005)

3. Cobb, B., Shenoy, P.P., Rumí, R.: Approximating probability density functions with mixtures of truncated exponentials. Statistics and Computing 16, 293–308 (2006)
4. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. Wiley Interscience, Chichester (2001)
5. Frank, E., Trigg, L., Holmes, G., Witten, I.H.: Technical note: Naive Bayes for regression. Machine Learning 41, 5–25 (2000)
6. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29, 131–163 (1997)
7. Gámez, J.A., Salmerón, A.: Predicción del valor genético en ovejas de raza manchega usando técnicas de aprendizaje automático. In: Actas de las VI Jornadas de Transferencia de Tecnología en Inteligencia Artificial, Paraninfo, pp. 71–80 (2005)
8. Jensen, F.V.: Bayesian networks and decision graphs. Springer, Heidelberg (2001)
9. Moral, S., Rumí, R., Salmerón, A.: Mixtures of truncated exponentials in hybrid Bayesian networks. In: Benferhat, S., Besnard, P. (eds.) ECSQARU 2001. LNCS (LNAI), vol. 2143, pp. 135–143. Springer, Heidelberg (2001)
10. Moral, S., Rumí, R., Salmerón, A.: Approximating conditional MTE distributions by means of mixed trees. In: Nielsen, T.D., Zhang, N.L. (eds.) ECSQARU 2003. LNCS (LNAI), vol. 2711, pp. 173–183. Springer, Heidelberg (2003)
11. Morales, M., Rodríguez, C., Salmerón, A.: Selective naive Bayes predictor using mixtures of truncated exponentials. In: Proceedings of the International Conference on Mathematical and Statistical Modelling (ICMSM'06) (2006)
12. Pearl, J.: Probabilistic reasoning in intelligent systems. Morgan Kaufmann, San Francisco (1988)
13. Pérez, A., Larrañaga, P., Inza, I.: Supervised classification with conditional Gaussian networks: Increasing the structure complexity from naive Bayes. International Journal of Approximate Reasoning 43, 1–25 (2006)
14. Quinlan, J.R.: Induction of decision trees. Machine Learning 1, 81–106 (1986)
15. Romero, V., Rumí, R., Salmerón, A.: Learning hybrid Bayesian networks using mixtures of truncated exponentials. International Journal of Approximate Reasoning 42, 54–68 (2006)
16. Rumí, R., Salmerón, A.: Approximate probability propagation with mixtures of truncated exponentials. International Journal of Approximate Reasoning (2007) (in press)
17. Rumí, R., Salmerón, A., Moral, S.: Estimating mixtures of truncated exponentials in hybrid Bayesian networks. Test 15, 397–421 (2006)

DENCLUE 2.0: Fast Clustering Based on Kernel Density Estimation

Alexander Hinneburg[1] and Hans-Henning Gabriel[2]

[1] Institute of Computer Science
Martin-Luther-University Halle-Wittenberg, Germany
hinneburg@informatik.uni-halle.de
[2] Otto-von-Guericke-University Magdeburg, Germany
Hans-Henning.Gabriel@web.de

Abstract. The DENCLUE algorithm employs a cluster model based on kernel density estimation. A cluster is defined by a local maximum of the estimated density function. Data points are assigned to clusters by hill climbing, i.e. points going to the same local maximum are put into the same cluster. A disadvantage of DENCLUE 1.0 is, that the used hill climbing may make unnecessary small steps in the beginning and never converges exactly to the maximum, it just comes close.

We introduce a new hill climbing procedure for Gaussian kernels, which adjusts the step size automatically at no extra costs. We prove that the procedure converges exactly towards a local maximum by reducing it to a special case of the expectation maximization algorithm. We show experimentally that the new procedure needs much less iterations and can be accelerated by sampling based methods with sacrificing only a small amount of accuracy.

1 Introduction

Clustering can be formulated in many different ways. Non-parametric methods are well suited for exploring clusters, because no generative model of the data is assumed. Instead, the probability density in the data space is directly estimated from data instances. Kernel density estimation [15,14] is a principled way of doing that task. There are several clustering algorithms, which exploit the adaptive nature of a kernel density estimate. Examples are the algorithms by Schnell [13] and Fukunaga [5] which use the gradient of the estimated density function. The algorithms are also described in the books by Bock [3] and Fukunaga [4] respectively. The DENCLUE framework for clustering [7,8] builds upon Schnells algorithm. There, clusters are defined by local maxima of the density estimate. Data points are assigned to local maxima by hill climbing. Those points which are assigned to the same local maximum are put into a single cluster.

However, the algorithms use directional information of the gradient only. The step size remains fixed throughout the hill climbing. This implies certain disadvantages, namely the hill climbing does not converges towards the local maximum, it just comes close, and the number of iteration steps may be large due to

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 70–80, 2007.

many unnecessary small steps in the beginning. The step size could be heuristically adjusted by probing the density function at several positions in the direction of the gradient. As the computation of the density function is relatively costly, such a method involves extra costs for step size adjustment, which are not guaranteed to be compensated by less iterations.

The contribution of this article is a new hill climbing method for kernel density estimates with Gaussian kernels. The new method adjusts the step size automatically at no additional costs and converges towards a local maximum. We prove this by casting the hill climbing as a special case of the expectation maximization algorithm. Depending on the convergence criterium, the new method needs less iterations as fixed step size methods. Since the new hill climbing can be seen as an EM algorithm, general acceleration methods for EM, like sparse EM [11] can be used as well. We also explore acceleration by sampling. Fast Density estimation [17] can be combined with our method as well but is not tested in this first study.

Other density based clustering methods beside DENCLUE, which would benefit from the new hill climbing, have been proposed by Herbin et al [6]. Variants of density based clustering are DBSCAN [12], OPTICS [1], and followup versions, which, however, do not use a probabilistic framework. This lack of foundation prevents the application of our new method there.

Related approaches include fuzzy c-means [2], which optimized the location of cluster centers and uses membership functions in a similar way as kernel functions are used by DENCLUE. A subtle difference between fuzzy c-means and DENCLUE is, that in c-means the membership grades of a point belonging to a cluster are normalized, s.t. the weights of a single data point for all clusters sum to one. This additional restriction makes the clusters competing for data points. DENCLUE does not have such restriction. The mountain method [16] also uses similar membership grades as c-means. It finds clusters by first discretizing the data space into a grid, calculates for all grid vertices the mountain function (which is comparable to the density up to normalization) and determines the grid vertex with the maximal mountain function as the center of the dominant cluster. After effects of the dominant cluster on the mountain function are removed, the second dominant cluster is found. The method iterates until the heights of the clusters drop below a predefined percentage of the dominant cluster. As the number of grid vertices grow exponentially in high dimensional data spaces, the method is limited to low dimensional data. Niche clustering [10] uses a non-normalized density function as fitness function for prototype-based clustering in a genetic algorithm. Data points with high density (larger than a threshold) are seen as core points, which are used to estimate scale parameters similar to the smoothing parameter h introduced in the next section.

The rest of the paper is structured as follows. In section 2, we briefly introduce the old DENCLUE framework and in section 3 we propose our new improvements for that framework. In section 4, we compare the old and the new hill climbing experimentally.

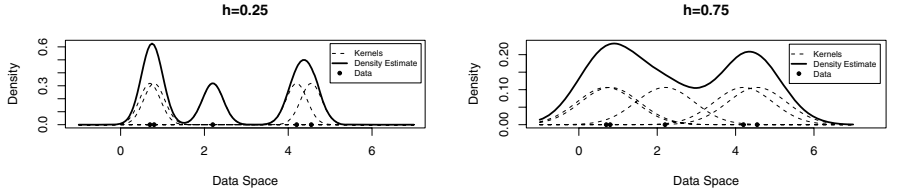


Fig. 1. Kernel density estimate for one-dimensional data and different values for the smoothing parameter h

2 DENCLUE 1.0 Framework for Clustering

The DENCLUE framework [8] builds on non-parametric methods, namely kernel density estimation. Non-parametric methods are not looking for optimal parameters of some model, but estimate desired quantities like the probability density of the data directly from the data instances. This allows a more direct definition of a clustering in contrast to 'parametric methods, where a clustering corresponds to an optimal parameter setting of some high-dimensional function. In the DENCLUE framework, the probability density in the data space is estimated as a function of all data instances $\boldsymbol{x}_t \in X \subset \mathbb{R}^d, d \in \mathbb{N},\ t = 1, \ldots, N$. The influences of the data instances in the data space are modeled via a simple kernel function, e.g. the Gaussian kernel $K(\boldsymbol{u}) = (2\pi)^{-\frac{d}{2}} \cdot \exp\left[-\frac{\boldsymbol{u}^2}{2}\right]$. The sum of all kernels (with suitable normalization) gives an estimate of the probability at any point $\boldsymbol{x}$ in the data space $\hat{p}(\boldsymbol{x}) = {}^1/_{(Nh^d)} \sum_{t=1}^{N} K\left(\boldsymbol{x}-\boldsymbol{x}_t/h\right)$. The estimate $\hat{p}(\boldsymbol{x})$ enjoys all properties like differentiability like the original kernel function. The quantity $h > 0$ specifies to what degree a data instance is smoothed over data space. When h is large, an instance stretches its influence up to more distant regions. When h is small, an instance effects only the local neighborhood. We illustrate the idea of kernel density estimation on one-dimensional data as shown in figure 1.

A clustering in the DENCLUE framework is defined by the local maxima of the estimated density function. A hill-climbing procedure is started for each data instance, which assigns the instance to a local maxima. In case of Gaussian kernels, the hill climbing is guided by the gradient of $\hat{p}(\boldsymbol{x})$, which takes the form

$$\nabla \hat{p}(\boldsymbol{x}) = \frac{1}{h^{d+2}N} \sum_{t=1}^{N} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_t}{h}\right) \cdot (\boldsymbol{x}_t - \boldsymbol{x}). \tag{1}$$

The hill climbing procedure starts at a data point and iterates until the density does not grow anymore. The update formula of the iteration to proceed from $\boldsymbol{x}^{(l)}$ to $\boldsymbol{x}^{(l+1)}$ is

$$\boldsymbol{x}^{(l+1)} = \boldsymbol{x}^{(l)} + \delta \frac{\nabla \hat{p}(\boldsymbol{x}^{(l)})}{\|\nabla \hat{p}(\boldsymbol{x}^{(l)})\|_2}. \tag{2}$$

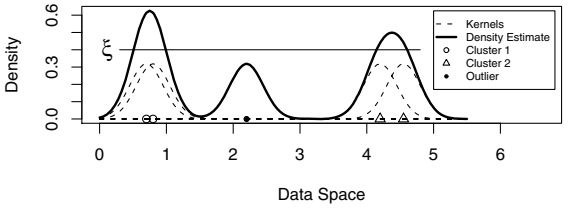


Fig. 2. Example of a DENCLUE clustering based on a kernel density estimate and a noise threshold ξ

The step size δ is a small positive number. In the end, those end points of the hill climbing iteration, which are closer than 2δ are considered, to belong to the same local maximum. Instances, which are assigned to the same local maximum, are put into the same cluster.

A practical problem of gradient based hill climbing in general is the adaptation of the step size. In other words, how far to follow the direction of the gradient? There are several general heuristics for this problem, which all need to calculate $\hat{p}(\boldsymbol{x})$ several times to decide a suitable step size.

In the presence of random noise in the data, the DENCLUE framework provides an extra parameter $\xi > 0$, which treats all points assigned to local maxima $\hat{\boldsymbol{x}}$ with $\hat{p}(\hat{\boldsymbol{x}}) < \xi$ as outliers. Figure 2 sketches the idea of a DENCLUE clustering.

3 DENCLUE 2.0

In this section, we propose significant improvements of the DENCLUE 1.0 framework for Gaussian kernels. Since the choice of the kernel type does not have large effects on the results in the typical case, the restriction on Gaussian kernels is not very serious. First, we introduce a new hill climbing procedure for Gaussian kernels, which adjust the step size automatically at no extra costs. The new method does really converge towards a local maximum. We prove this property by casting the hill climbing procedure as an instance of the expectation maximization algorithm. Last, we propose sampling based methods to accelerate the computation of the kernel density estimate.

3.1 Fast Hill Climbing

The goal of a hill climbing procedure is to maximize the density $\hat{p}(\boldsymbol{x})$. An alternative approach to gradient based hill climbing is to set the first derivative of $\hat{p}(\boldsymbol{x})$ to zero and solve for $\boldsymbol{x}$. Setting (1) to zero and rearranging we get

$$\boldsymbol{x} = \frac{\sum_{t=1}^{N} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_t}{h}\right)\boldsymbol{x}_t}{\sum_{t=1}^{N} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_t}{h}\right)} \tag{3}$$

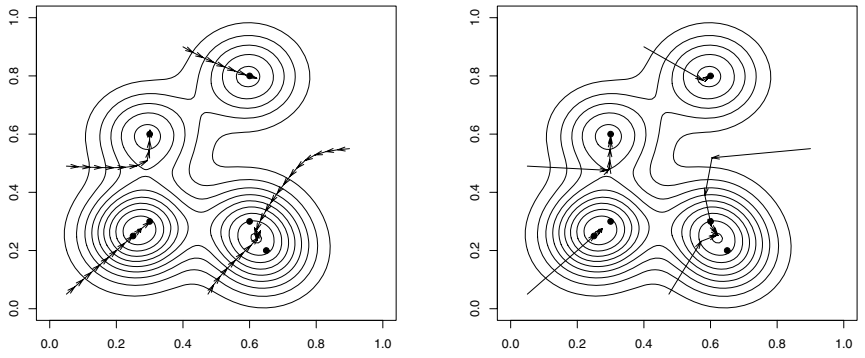


Fig. 3. (Left) Gradient hill climbing as used by DENCLUE 1.0, (right) Step size adjusting hill climbing used by DENCLUE 2.0

Obviously, this is not a solution for $\boldsymbol{x}$, since the vector is still involved into the righthand side. Since $\boldsymbol{x}$ influences the righthand side only through the kernel, the idea is to compute the kernel for some fixed $\boldsymbol{x}$ and update the vector on the lefthand side according to formula (3). This give a new iterative procedure with the update formula

$$\boldsymbol{x}^{(l+1)} = \frac{\sum_{t=1}^{N} K\left(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h}\right) \boldsymbol{x}_t}{\sum_{t=1}^{N} K\left(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h}\right)} \tag{4}$$

The update formula can be interpreted as a normalized and weighted average of the data points and the weights of the data points depend on the influence of their kernels on the current $\boldsymbol{x}^{(l)}$. In order to see that the new update formula makes sense it is interesting to look at the special case $N = 1$. In that case, the estimated density function consists just of a single kernel and the iteration jumps after one step to $\boldsymbol{x}^1$, which is the maximum.

The behavior of DENCLUEs 1.0 hill climbing and the new hill climbing procedure is illustrated in figure 3. The figure shows that the step size of the new procedure is adjusted to the shape of the density function. On the other hand, an iteration of the new procedure has the same computational costs as one of the old gradient based hill climbing. So, adjusting the step size comes at no additional costs. Another difference is, that the hill climbing of the new method really converges towards a local maximum, while the old method just comes close.

Since the new method does not need the step size parameter δ, the assignment of the instances to clusters is done in a new way. The problem is to define a heuristic, which automatically adjusts to the scale of distance between the converged points.

A hill climbing is started at each data point $\boldsymbol{x}_t \in X$ and iterates until the density does not change much, i.e. $[\hat{f}(\boldsymbol{x}_t^{(l)}) - \hat{f}(\boldsymbol{x}_t^{(l-1)})] / \hat{f}(\boldsymbol{x}_t^{(l)}) \leq \epsilon$. An end point reached by the hill climbing is denoted by $\boldsymbol{x}_t^* = \boldsymbol{x}_t^{(l)}$ and the sum of the k last

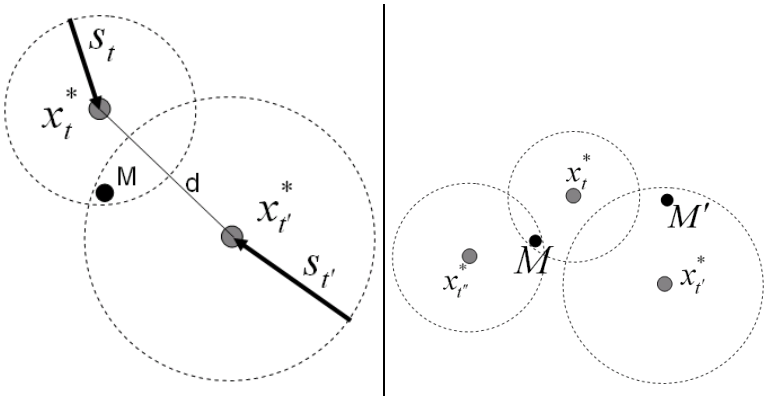


Fig. 4. (Left) Assignment to a local maximum, (right) Ambiguous assignment. The points M and M' denote the true but unknown local maxima.

step sizes is $s_t = \sum_{i=1}^{k} \|\boldsymbol{x}_t^{(l-i+1)} - \boldsymbol{x}_t^{(l-i)}\|_2$. The integer k is parameter of the heuristic. We found that $k = 2$ worked well for all experiments. Note, that the number of iterations may vary between the data points, however, we restricted the number of iterations to be larger than k. For appropriate $\epsilon > 0$, it is safe to assume that the end points $\boldsymbol{x}_t^*$ are close to the respective local maxima. Typically, the step sizes are strongly shrinking before the convergence criterium is met. Therefore, we assume that the true local maximum is within a ball around $\boldsymbol{x}_t^*$ of radius s_t. Thus, the points belonging to the same local maximum have end points $\boldsymbol{x}_t^*$ and $\boldsymbol{x}_{t'}^*$, which are closer than $s_t + s_{t'}$. Figure 4 left illustrates that case.

However, there might exists rare cases, when such an assignment is not unique. This happens when for three end points $\boldsymbol{x}_t^*, \boldsymbol{x}_{t'}^*$ and $\boldsymbol{x}_{t''}^*$ hold the following conditions $\|\boldsymbol{x}_t^* - \boldsymbol{x}_{t'}^*\| \leq s_t + s_{t'}$ and $\|\boldsymbol{x}_t^* - \boldsymbol{x}_{t''}^*\| \leq s_t + s_{t''}$ but not $\|\boldsymbol{x}_{t'}^* - \boldsymbol{x}_{t''}^*\| \leq s_{t'} + s_{t''}$. In order to solve the problem, the hill climbing is continued for all points, which are involved in such situations, until the convergence criterium is met for some smaller ϵ (a simple way to reduce ϵ is multiply it with a constant between zero and one). After convergence is reached again, the ambiguous cases are rechecked. The hill climbing is continued until all such cases are solved. Since further iterations causes the step sizes to shrink the procedure will stop at some point. The idea is illustrated in figure 4 right.

However, until now it is not clear why the new hill climbing procedure converges towards a local maximum. In the next section, we prove this claim.

3.2 Reduction to Expectation Maximization

We prove the convergence of the new hill climbing method by casting the maximization of the density function as a special case of the expectation maximization framework [9]. When using the Gaussian kernel we can rewrite the kernel

density estimate $\hat{p}(\boldsymbol{x})$ in the form of a constrained mixture model with Gaussian components

$$p(\boldsymbol{x}|\boldsymbol{\mu},\sigma) = \sum_{t=1}^{N} \pi_t \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma) \tag{5}$$

and the constraints $\pi_t = 1/N$, $\boldsymbol{\mu}_t = \boldsymbol{x}_t$ ($\boldsymbol{\mu}$ denotes a vector consisting of all concatenated $\boldsymbol{\mu}_t$), and $\sigma = h$. We can think of $p(\boldsymbol{x}|\boldsymbol{\mu},\sigma)$ as a likelihood of $\boldsymbol{x}$ given the model determined by $\boldsymbol{\mu}$ and σ. Maximizing $\log p(\boldsymbol{x}|\boldsymbol{\mu},\sigma)$ wrt. $\boldsymbol{x}$ is not possible in a direct way. Therefore, we resort to the EM framework by introducing a hidden bit variable $\boldsymbol{z} \in \{0,1\}^N$ with $\sum_{t=1}^{N} z_t = 1$ and

$$z_t = \begin{cases} 1 & \text{if the density at } \boldsymbol{x} \text{ is explained by } \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma) \text{ only} \\ 0 & \text{else} \end{cases}. \tag{6}$$

The complete log-likelihood is $\log p(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{\mu},\sigma) = \log p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\mu},\sigma)p(\boldsymbol{z})$ with $p(\boldsymbol{z}) = \prod_{t=1}^{N} \pi_t^{z_t}$ and $p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\mu},\sigma) = \prod_{t=1}^{N} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma)^{z_t}$.

In contrast to generative models, which use EM to determine parameters of the model, we maximize the complete likelihood wrt. $\boldsymbol{x}$. The EM-framework ensures that maximizing the complete log-likelihood maximizes the original log-likelihood as well. Therefore, we define the quantity

$$\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)}) = E[\log p(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{\mu},\sigma)|\boldsymbol{\mu},\sigma,\boldsymbol{x}^{(l)}] \tag{7}$$

In the E-step the expectation $\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)})$ is computed wrt. to $\boldsymbol{z}$ and $\boldsymbol{x}^{(l)}$ is put for $\boldsymbol{x}$, while in the M-step $\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)})$ is taken as a function of $\boldsymbol{x}$ and maximized. The E-step boils down to compute the posterior probability for the z_t:

$$E[z_t|\boldsymbol{\mu},\sigma,\boldsymbol{x}^{(l)}] = p(z_t = 1|\boldsymbol{x}^{(l)},\boldsymbol{\mu},\sigma) \tag{8}$$

$$= \frac{p(\boldsymbol{x}^{(l)}|z_t = 1,\boldsymbol{\mu},\sigma)p(z_t = 1|\boldsymbol{\mu},\sigma)}{\sum_{t=1}^{N} p(\boldsymbol{x}^{(l)}|z_t = 1,\boldsymbol{\mu},\sigma)p(z_t = 1|\boldsymbol{\mu},\sigma)} \tag{9}$$

$$= \frac{1/N \cdot \mathcal{N}(\boldsymbol{x}^{(l)}|\boldsymbol{\mu}_t,\sigma)}{\sum_{t'=1}^{N} 1/N \cdot \mathcal{N}(\boldsymbol{x}^{(l)}|\boldsymbol{\mu}_{t'},\sigma)} \tag{10}$$

$$= \frac{1/N \cdot K(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h})}{\hat{p}(\boldsymbol{x}^{(l)})} = \theta_t \tag{11}$$

In the M-step, z_t is replaced by the fixed posterior θ_t, which yields $\mathcal{Q}(\boldsymbol{x}|\boldsymbol{x}^{(l)}) = \sum_{t=1}^{N} \theta_t[\log 1/N + \log \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_t,\sigma)]$. Computing the derivative wrt. $\boldsymbol{x}$ and setting it to zero yields $\sum_{t=1}^{N} \theta_t \sigma^{-2}(\boldsymbol{x} - \boldsymbol{\mu}_t) = 0$ and thus

$$\boldsymbol{x}^{(l+1)} = \frac{\sum_{t=1}^{N} \theta_t \mu_t}{\sum_{t=1}^{N} \theta_t} = \frac{\sum_{t=1}^{N} K(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h})\boldsymbol{x}_t}{\sum_{t=1}^{N} K(\frac{\boldsymbol{x}^{(l)} - \boldsymbol{x}_t}{h})} \tag{12}$$

By starting the EM with $\boldsymbol{x}^{(0)} = \boldsymbol{x}_t$ the method performs an iterative hill climbing starting at data point $\boldsymbol{x}_t$.

3.3 Sampling Based Acceleration

As the hill climbing procedure is a special case of the expectation maximization algorithm, we can employ different general acceleration techniques known for EM to speed up the the DENCLUE clustering algorithm.

Most known methods for EM, try to reduce the number of iterations needed until convergence [9]. Since the number of iterations is typically quite low, that kind of techniques yield no significant reduction for the clustering algorithm.

In order to speed up the clustering algorithm, the costs for the iterations itself should be reduced. One option is sparse EM [11], which still converges to the true local maxima. The idea is to freeze small posteriors for several iterations, so only the $p\%$ largest posteriors are updated in each iteration. As the hill climbing typically needs only a few iterations we modify the hill climbing starting at the single point $\boldsymbol{x}^{(0)}$ as follows. All kernels $K(\frac{\boldsymbol{x}^{(0)}-\boldsymbol{x}_t}{h})$ are determined in the initial iteration and $\boldsymbol{x}^{(1)}$ is determined as before. Let be U the index set of the $p\%$ largest kernels and L the complement. Then, in the next iterations the update formula is modified to

$$\boldsymbol{x}^{(l+1)} = \frac{\sum_{t\in U} K(\frac{\boldsymbol{x}^{(l)}-\boldsymbol{x}_t}{h})\boldsymbol{x}_t + \sum_{t\in L} K(\frac{\boldsymbol{x}^{(0)}-\boldsymbol{x}_t}{h})\boldsymbol{x}_t}{\sum_{t\in U} K(\frac{\boldsymbol{x}^{(l)}-\boldsymbol{x}_t}{h}) + \sum_{t\in L} K(\frac{\boldsymbol{x}^{(0)}-\boldsymbol{x}_t}{h})} \tag{13}$$

The index set U and L can be computed by sorting. The disadvantage of the method is, that the first iteration is still the same as in the original EM.

The original hill climbing converges towards a true local maximum of the density function. However, we does not need the exact position of such a maximum. It is sufficient for the clustering algorithm, that all points of a cluster converge to the same local maximum, regardless where that location might be. In that light, it makes sense to simplify the original density function by reducing the data set to a set of $p\%$ representative points. That reduction can be done in many ways. We consider here random sampling and k-means. So the number of points N is reduced to a much smaller number of representative points N', which are used to construct the density estimate.

Note that random sampling has much smaller costs as k-means. We investigate in the experimental section, whether the additional costs by k-means pay off by less needed iterations or by cluster quality.

4 Experimental Evaluation

We compared the new step size adjusting (SSA) hill climbing method with the old fixed step size hill climbing. We used synthetic data with normally distributed 16-dimensional clusters with uniformly distributed centers and approximately same size. Both methods are tuned to find the perfect clustering in the most efficient way. The total sum of numbers of iterations for the hill climbings of all data points is plotted versus the number of data points. SSA was run with different values for ϵ, which controls the convergence criterium of SSA. Figure 5

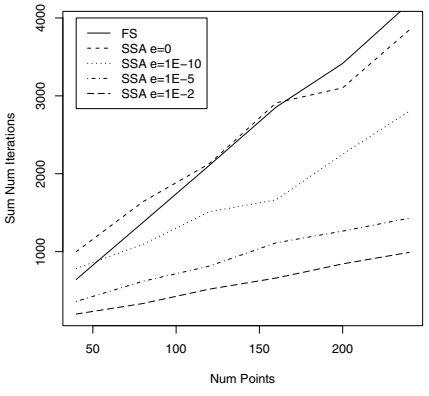


Fig. 5. Number of data points versus the total sum of numbers of iterations

clearly shows that SSA ($\epsilon = 0.01$) needs only a fraction of the number of iterations of FS to achieve the same results. The costs per iterations are the same for both methods.

Next, we tested the influence of different sampling methods on the computational costs. Since the costs per iteration differ for sparse EM, we measure the costs in number of kernel computations versus sample size. Figure 6(left) shows that sparse EM is more expensive than random sampling and k-means based data reduction. The difference between the two latter methods is negligible, so the additional effort of k-means during the data reduction does not pay off in less computational costs during the hill climbing. For sample size 100% the methods converge to the original SSA hill climbing.

For random sampling, we tested sample size versus cluster quality measured by normalized mutual information (NMI is one if the perfect clustering is found). Figure 6(right) shows that the decrease of cluster quality is not linear in sample size. So, a sample of 20% is still sufficient for a good clustering when the dimensionality is $d = 16$. Larger dimensionality requires larger samples as well as more smoothing (larger h), but the clustering can still be found.

In the last experiment, we compared SSA, its sampling variants, and k-means with the optimal k on various real data sets from the machine learning repository wrt. cluster quality. Table 1 shows average values of NMI with standard deviation for k-means and sampling, but not for SSA which is a deterministic algorithm.

SSA has better or comparable cluster quality as k-means. The sampling variants degrade with smaller sample sizes $(0.8, 0.4, 0.2)$, but k-means based data reduction suffers much less from that effect. So, the additional effort of k-means based data reduction pays off in cluster quality.

In all experiments, the smoothing parameter h was tuned manually. Currently, we are working on methods to determine that parameter automatically. In conclusion, we proposed a new hill climbing method for kernel density functions,

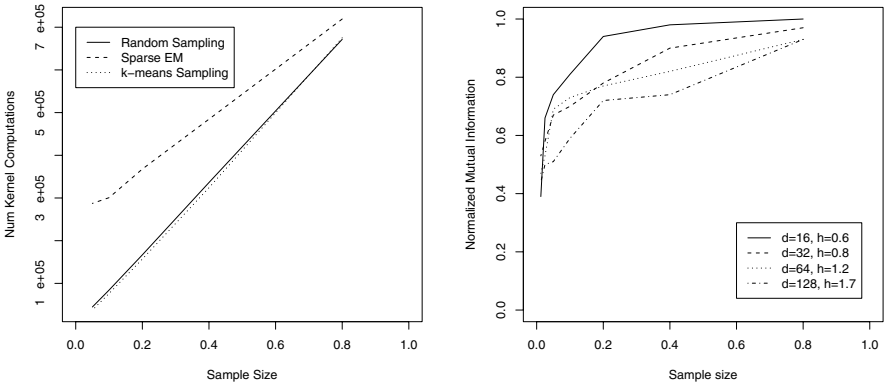


Fig. 6. (Left) Sample size versus number of kernel computations, (right) sample size versus cluster quality (normalized mutual information, NMI).

Table 1. NMI values for different data and methods, the first number in the three rightmost columns shows the sample size

	k-means	**SSA**	**Random Sampling**	**Sparse EM**	**k-means Sampling**
iris	0.69±0.10	0.72	0.8: 0.66±0.05	0.8: 0.68±0.06	0.8: 0.67±0.06
			0.4: 0.63±0.05	0.4: 0.60±0.06	0.4: 0.65±0.07
			0.2: 0.63±0.06	0.2: 0.50±0.04	0.2: 0.64±0.07
ecoli	0.56±0.05	0.67	0.8: 0.65±0.02	0.8: 0.66±0.00	0.8: 0.65±0.02
			0.4: 0.62±0.06	0.4: 0.61±0.00	0.4: 0.65±0.04
			0.2: 0.59±0.06	0.2: 0.40±0.00	0.2: 0.65±0.03
wine	0.82±0.14	0.80	0.8: 0.71±0.06	0.8: 0.72±0.07	0.8: 0.70±0.11
			0.4: 0.63±0.10	0.4: 0.63±0.00	0.4: 0.70±0.05
			0.2: 0.55±0.15	0.2: 0.41±0.00	0.2: 0.58±0.21

which really converges towards a local maximum and adjusts the step size automatically. We believe, that our new technique has some potential for interesting combinations with parametric clustering methods.

References

1. Ankerst, M., Breunig, M.M., Kriegel, H.-P., Sander, J.: Optics: Ordering points to identify the clustering structure. In: Proceedings SIGMOD'99, pp. 49–60. ACM Press, New York (1999)
2. Bezdek, J.: Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. Kluwer Academic Publishers, Dordrecht (1999)
3. Bock, H.H.: Automatic Classification. Vandenhoeck and Ruprecht (1974)
4. Fukunaga, K.: Introduction to Statistical Pattern Recognition. Academic Press, London (1990)
5. Fukunaga, K., Hostler, L.: The estimation of the gradient of a density function, with application in pattern recognition. IEEE Trans. Info. Thy. 21, 32–40 (1975)

6. Herbin, M., Bonnet, N., Vautrot, P.: Estimation of the number of clusters and influence zones. Pattern Recognition Letters 22, 1557–1568 (2001)
7. Hinneburg, A., Keim, D.: An efficient approach to clustering in large multimedia databases with noise. In: Proceedings KDD'98, pp. 58–65. AAAI Press (1998)
8. Hinneburg, A., Keim, D.A.: A general approach to clustering in large databases with noise. Knowledge and Information Systems (KAIS) 5(4), 387–415 (2003)
9. McLachlan, G.J., Krishnan, T.: EM Algorithm and Extensions. Wiley, Chichester (1997)
10. Nasraoui, O., Krishnapuram, R.: The unsupervised niche clustering algorithm: extension tomultivariate clusters and application to color image segmentation. In: IFSA World Congress and 20th NAFIPS International Conference, vol. 3 (2001)
11. Neal, R.M., Hinton, G.E.: A view of the em algorithm that justifies incremental, sparse, and other variants. In: Learning in graphical models, pp. 355–368. MIT Press, Cambridge (1999)
12. Sander, J., Ester, M., Kriegel, H.-P., Xu, X.: Density-based clustering in spatial databases: The algorithm gdbscan and its applications. Data Mining and Knowledge Discovery 2(2), 169–194 (1997)
13. Schnell, P.: A method to find point-groups. Biometrika 6, 47–48 (1964)
14. Scott, D.: Multivariate Density Estimation. Wiley, Chichester (1992)
15. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapman & Hall (1986)
16. Yager, R., Filev, D.: Approximate clustering via the mountain method. IEEE Transactions on Systems, Man and Cybernetics 24(8), 1279–1284 (1994)
17. Zhang, T., Ramakrishnan, R., Livny, M.: Fast density estimation using cf-kernel for very large databases. In: Proceedings KDD'99, pp. 312–316. ACM Press, New York (1999)

Visualising the Cluster Structure of Data Streams

Dimitris K. Tasoulis[1], Gordon Ross[2], and Niall M. Adams[2]

[1] Institute for Mathematical Sciences
Imperial College London, South Kensington Campus
London SW7 2PG, United Kingdom
[2] Department of Mathematics
Imperial College London, South Kensington Campus
London SW7 2PG, United Kingdom
{d.tasoulis,gordon.ross,n.adams}@imperial.ac.uk

Abstract. The increasing availability of streaming data is a consequence of the continuing advancement of data acquisition technology. Such data provides new challenges to the various data analysis communities. Clustering has long been a fundamental procedure for acquiring knowledge from data, and new tools are emerging that allow the clustering of data streams. However the dynamic, temporal components of streaming data provide extra challenges to the development of stream clustering and associated visualisation techniques. In this work we combine a streaming clustering framework with an extension of a static cluster visualisation method, in order to construct a surface that graphically represents the clustering structure of the data stream. The proposed method, OpticsStream, provides intuitive representations of the clustering structure as well as the manner in which this structure changes through time.

1 Introduction

Advances in technology have resulted in an increasing number of application areas generating streaming data, that is, data obtained by observation of multiple indefinitely long and time-evolving sequences. These applications areas include astronomy and earth sciences, telecommunications and network monitoring. Information visualisation techniques have provided a means to help the exploration of large data sets [4]. As such there is a need to develop visual data analysis methods that can deal successfully with the challenges arising from the dynamically changing nature of streaming data.

Clustering is the process of partitioning a data set into homogeneous subsets called clusters. Since the publication of the first comprehensive study of clustering algorithms [14], these methods have been applied to a wide variety of fields, ranging from text mining [6] to genomics [7].

Cluster visualisation for two dimensional data is readily achieved using simple scatter plots. To handle higher dimensional data most methods try to determine two- or three-dimensional projections of the data that retain certain properties of

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 81–92, 2007.

the high-dimensional clusters. Dendrograms, produced by hierarchical clustering algorithms, are a popular visualisation techniques despite the quadratic time required to construct them [9]. However such techniques become difficult to apply in large, high-dimensional data sets, particularly when the clusters are not clearly separated. More sophisticated techniques have been developed to address such issues [3,8].

Although there has been some effort [10] to extend cluster visualisation to streaming data, the focus has been applications with a relatively stable, periodically updated environment. In detail they assume a permanent database to which insertions and deletion operations occur. Moreover, they are designed with the assumption that all the data can be stored and retrieved whenever it is required. Furthermore they do not incorporate any forgetting mechanism. As such, almost all the foundational streaming data problems are not addressed by such approaches.

In this paper we present an extension of the OPTICS algorithm [3] to the streaming data model. OPTICS uses a density identification method to create a one-dimensional cluster-ordering of the data. We will show that embodying this idea in a stream clustering method can provide new insights into both the current clustering structure and the structure evolution. This is a critical issue in this area since the basic assumption of a data stream model is the dynamically changing nature of the streams. Thus, such an algorithm should have the capacity to adapt its visualisation medium to rapidly changing dynamics of the sequences. Finally, scalability in the number of sequences is becoming increasingly desirable, as data collection technology develops.

The paper proceeds as follows: the next section provides a description of cluster visualisation in static data sets. Stream clustering is described in Section 3, along with the proposed visualisation method, called *OpticsStream*. Next, in Section 4, we demonstrate the behaviour of the OpticsStream with an experimental analysis of both artificial and real data sets. The paper concludes with a discussion in Section 5.

2 Visualising Clusters in Static Data

Unlike many other procedures, the OPTICS algorithm [3] (Ordering Points To Identify the Clustering Structure) does not produce an explicit clustering of a data set, but instead creates an augmented ordering of the data representing its density-based clustering structure. For medium sized data sets, this cluster-ordering can be represented graphically, a fact that allows interactive exploration of the intrinsic clustering structure and thereby offering additional insight into the distribution of the data

Consider a data set $X \subset \mathbb{R}^d$ of n vectors (objects), and a metric distance function $dist : X \times X \to \mathbb{R}$. Additionally, for each object $p \in X$ and a value $\epsilon \in \mathbb{R}$, we define the set $N_\epsilon(p) = \{q \in X | dist(p, q) \leqslant \epsilon\}$, as the ϵ-neighbourhood of p. ϵ is a user defined parameter closely related to the application, the distance

metric and the dimensionality of the data. For more details about reasonable values for this refer to [12].

The OPTICS point ordering is constructed using two quantities, the *"core-level"* and the *"reachability-distance"*, that are computed for each object p in the database. These quantities are defined as follows:

Definition 1. *(core-level) Let $\epsilon \in \mathbb{R}, \mu \in \mathbb{N}$, be user defined parameters and $dist_\mu(p)$ be the distance from p to its μ-nearest neighbour. The core-level of p, $CLev(p)$ is defined as:*

$$CLev(p) \begin{cases} \infty & if \quad |N_\epsilon(p)| < \mu \\ dist_\mu(p) & otherwise \end{cases}$$

Where $|N_\epsilon(p)|$ is the number of objects from the database in the ϵ-neighbourhood of p. μ is the user defined that combined with value of ϵ, separates the data as outliers or not. If in the ϵ-neighbourhood around a point less than μ points reside then this corresponds to an outlier [12]. Next we define the *"reachability-distance"*, for each pair of objects in the database.

Definition 2. *(reachability) For two objects p, q in the database the reachability of p wrt q is defined as $RDist(p,q) = \max\{CLev(p), dist(p,q)\}$.*

Under these definition the cluster ordering of the data is constructed through the following definition:

Definition 3. *(cluster ordering) Let $\epsilon \in \mathbb{R}, \mu \in \mathbb{N}$, and CO be a totally ordered permutation of the n objects of X. Each $o \in X$, is assigned the additional attributes $Pos(o)$, $Core(o)$, and $Reach(o)$, where $Pos(o)$, symbolizes the position of o in CO. The ordering CO is called a cluster ordering wrt ϵ and μ if the following three conditions hold:*

1. $\forall p \in CO : Core(p) = CLev(p)$
2. $\forall o, x, y \in CO : Pos(x) < Pos(o) \wedge Pos(y) > Pos(o) \Rightarrow RDist(x,y) \geqslant RDist(o,x)$
3. $\forall p \in CO : Reach(p) = \inf\{RDist(p,o) | o \in CO \quad \text{and} \quad Pos(o) < Pos(p)\}$.

In the cluster ordering defined this way each object is positioned in the ordering so that it has minimum reachability distance to all preceding objects in the ordering. The cluster-ordering of a data set can be represented and understood graphically. In principle, one can see the clustering structure of a data set if the *reachability-distance* values r are plotted for each object against the corresponding cluster-ordering CO. An example of a very simple 2-dimensional data set is depicted in Fig. 1(a). The CO for this dataset is depicted in Fig. 1(b), where in the y-axis correpsonds to the reachability distance. A detailed description of the workings of the OPTICS algorithm is given by [3].

3 Stream Clustering

There are a large number of highly efficient and effective clustering methods [9]. However, most are *batch* methods originally designed for static data and therefore relying on the assumption that data are available in a permanent memory

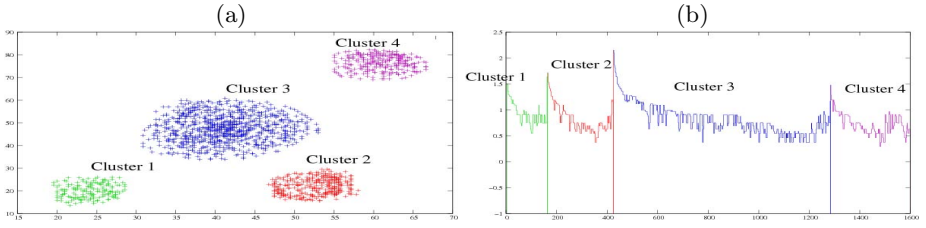


Fig. 1. Reachability Distance ordering (right) of a 2-dimensional data set with 4 clusters (left)

structure from which information can be retrieved at any time. Streaming data is more demanding, requiring a one-pass scan of the data. Thus, many standard clustering methods are not directly applicable to streaming data. More recent approaches have focused precisely on the streaming data model – accommodating both dynamics and the memory limitation issues [5,13].

There are very few methods for stream clustering visualisation. One example is described by [1], where so-called velocity density estimation is proposed to provide a visual diagnosis of changes in a data stream. Here, the idea is to estimate the rate of change ("velocity") of a kernel density estimate at each spatial location. This is obviously an extremely computer-intensive procedure. Moreover, to make the method applicable in greater than two dimensions, a projection technique is proposed, which increases the computational complexity of the algorithm still further.

A widely used and important concept in stream clustering is *micro-clusters*. These are quantities that try to summarise the data arriving over a continuous stream. In the next paragraph we present a micro-cluster framework.

3.1 The Micro-clustering Framework

Among the various models developed for data stream clustering the micro-clustering framework has been successfully employed by various different algorithms [2,5]. In this framework the weight of each data point in the stream decreases exponentially with time t via a fading function $T_\lambda(t) = 2^{-\lambda t}$, where $\lambda > 0$. The parameter λ control the rate that historic data is down-weighted. The smaller the value of λ, the greater importance is given to historical data. To this end we can extend both the notions of *"core-level"* and *"reachability"* used in static clustering algorithms [3,12] to the data stream setting.

For a spatio-temporal data set $X' = \{\{x_1, t_1\}, \ldots, \{x_n, t_n\}\}$, where x_i is a vector and t_i is the corresponding time-stamp, a micro-cluster is defined by the quantities w, c, r, which attempt to summarise information about the data density of a particular area. Two distinct types of micro-clusters are considered, based on the values of r, w, and the additional user-defined parameters ϵ and μ. If $r < \epsilon$ and $w > \mu$, then the micro-cluster is considered to be a *core-micro-cluster*, that accounts for a "dense" region of the data. Otherwise the micro-cluster

is called an *outlier-micro-cluster*, that accounts for less dense regions of data. Formally we can define these micro-clusters as follows:

Definition 4. *(core-micro-cluster) A micro-cluster* $MC_t(w, c, r)$ *is defined as a core-micro-cluster* $CMC_t(w, c, r)$ *at time* t *for a group of streaming points* $\{x_i, t_i\}$, $i = 1, \ldots, n$, *and parameters* ϵ, μ *when* $w \geqslant \mu$ *and* $r \geqslant \epsilon$. *Where* $w = \sum_{i=1}^{n} T_\lambda(t_i)$ *be the micro-cluster's weight,* $c = \frac{\sum_{i=1}^{n} x_i T_\lambda(t_i)}{w}$, *be its center and* r *its (weighted) mean radius,* $r = \sum_{i=1}^{n} \frac{T_\lambda(t_i)\|c - x_i\|}{w}$.

The intuition behind the micro-clustering framework is to maintain a description of the data streams by a list of micro-clusters. As data points are instantiated at each time step from the stream, they are *merged* to their nearest micro-clusters provided that they are sufficiently close. Otherwise new micro-clusters are spawned in order to merge possible future similar data, and thus capture the density structure of the stream. The quantities w, c, r, can be incrementally computed for each micro-cluster. Consider a micro-cluster for which no points were merged between time step t_p and the current time t_c, where the point x_c is being considered for merging. Thus $w_{t_c} = 2^{-\lambda(t_c - t_p)} w_p$, where w_p is the weight value at time t_p. Incrementally computing c, r involves the use of two quantities, $CF1_t = \{\sum_{i=1}^{n} x_{i,j} T_\lambda(t_i)\}$, for each coordinate j of the data point $x_i \in \mathbb{R}^d$, and $CF2_t = \{\sum_{i=1}^{n} x_{i,j}^2 T_\lambda(t_i)\}$. The quantities $CF1_t, CF2_t$ can be incrementally computed at time t_c as $CF1_{t_c} = 2^{-\lambda(t_c - t_p)} CF1_{t_c} + x_c$, and $CF2_{t_c} = 2^{-\lambda(t_c - t_p)} CF2_{t_p} + x_c^2$.

Note that the weight of core-micro-clusters must be greater than or equal to μ and the radius must be less than or equal to ϵ, in order to represent "dense" regions of the data space.

In an evolving data stream the role of clusters and outliers, that is points that do not participate in clusters, often exchanges. To compensate for this phenomenon, two types of micro-clusters are used [5]:

- *Potential core-micro-clusters*, when $w_c \leqslant \beta\mu$ and $r \leqslant \epsilon$,
- *Outlier-micro-clusters*, when $w_c > \beta\mu$ and $r > \epsilon$,

where β is a user defined parameter. As described above, the difference between these micro-clusters relates to the constraints on the weight and the radius of each micro-cluster. Maintaining two lists; one for the potential core-micro-clusters, and one for outlier-micro-clusters, and updating them on-line can provide a density description of the data space, that can be further queried to provide knowledge of the clustering structure. These two lists are maintained using the following procedure:

Procedure **ListMaintain**

1. Initialise two lists PL, OUL; one for the potential core-micro-clusters, and the other for the outlier-micro-clusters.
2. Each time a new point $p = \{x, t\}$ arrives do one of the following:
   (a) Attempt to merge p into its nearest potential core-micro-cluster c_p: If the resulting micro-cluster has a radius $r > \epsilon$ then the merge is omitted.

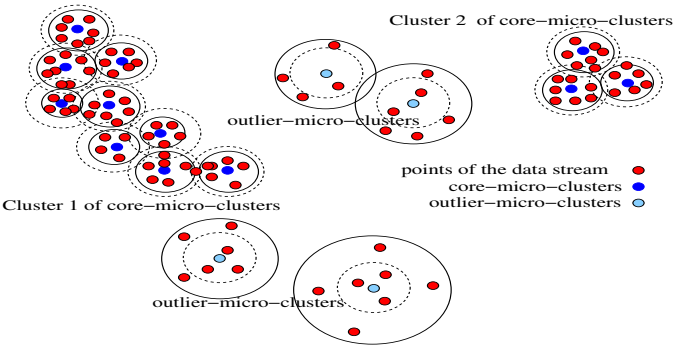


Fig. 2. An example of the result of the micro-clustering framework

(b) Attempt to merge p into its nearest outlier-micro-cluster o_p: if $r > \epsilon$ the merge is omitted. Otherwise, if the subsequent weight w of o_p exceeds μ, then move o_p to PL.
(c) A new outlier-micro-cluster is created, centered at p.

3. Periodically prune from PL and OUL those micro-clusters for which $w_c \leqslant \beta\mu$ and $w_c \leqslant \xi$.

Periodic pruning of the micro-clusters can occur sufficiently often such that the available memory in not exceeded. [5] proposes to conduct this pruning based on a so called *time-span* parameter of potential core-micro-clusters. Note also that ξ is a user-defined parameter that determines the lower limit for the weight of an outlier-micro-clusters before it is pruned, and its value can be connected with T_p [5].

Following the procedure outlined above, the micro-clusters maintained on-line capture the dense areas of data streams. An example is demonstrated in Fig. 2. Around each micro-cluster the ϵ area is depicted with a dashed circle, while the computed radius r of each micro-cluster is depicted with a solid circle.

3.2 Stream Cluster Visualisation

We now propose a stream clustering visualisation methodology. This approach can potentially operate in a real-time environment and can produce a time-sensitive map representing the clustering structure in an understandable format. To achieve this, the OPTICS methodology and the micro-clustering framework described above are combined, to provide a 3-dimensional plot that depicts the evolution of the stream cluster structure over time.

In short, we apply the concepts behind the OPTICS algorithm to the potential core-micro-cluster list, translated to the streaming data context. First, we need to define the core-micro-cluster neighbourhood:

Definition 5. *(Micro-cluster neighbourhood) Let $\epsilon \in \mathbb{R}$, be a user defined parameter and PL a potential core-micro-cluster list. Then for a potential core-micro-cluster c_p, we define the micro-cluster neighbourhood of c_p, as*

$$N(c_p) = \{c_q \in PL | dist(c_p, c_q) \leqslant 3.0\epsilon\}.$$

The function $dist(c_p, c_q)$*, returns the Euclidean distance between the centers of* c_p *and* c_q*.*

Note that the distance between the centers of two neighbouring micro-clusters is required to be less than 3 times the ϵ parameter. Intuitively, this considers micro-clusters to constitute contiguous high density areas when the distance between the ϵ radius spheres around them is less than ϵ. Although this is a crude way of defining neighborhoods, it is possible to use other measures that take into consideration the radius of each micro-cluster. We can also extend the definition of core-level distance to the micro-cluster framework as follows:

Definition 6. *(Micro-cluster core-level) Let* $\epsilon \in \mathbb{R}, \beta \in \mathbb{R}, \mu \in \mathbb{N}$*. The core-level of* c_p*,* $CLev(c_p)$ *is defined as:*

$$CLev(c_p) = \text{radius} \quad \text{of} \quad c_p$$

The difference here is that there is no need to compute core-level of a micro-cluster from the neighbourhood of c_p, as such information has already been incorporated into the radius of c_p. Both the definitions of micro-cluster reachability and ordering remain the same as definitions 2 and 3, respectively. Assuming the existence of a potential micro-cluster list PL we can construct an order list OL from it by applying the following algorithm:

Procedure **StreamOptics**

1. While there is still a micro-cluster c_p in PL that has a neighbourhood size $|N(c_p)| > 1$, initialize a list S of all the micro-clusters in $N(c_p)$.
2. Remove c_p from PL and add to OL.
3. Remove all micro-clusters in $N(c_p)$ from PL.
4. For each c_l in S, compute $RDist(c_l, c_p)$.
5. For each c_l in S, insert all the micro-clusters in $N(c_l)$ to S.
6. Remove from PL all the micro-clusters in S.
7. Remove the object c_l from S with the smallest $RDist(c_l, c_p)$, and insert it OL , until S is empty.

The changes which occur at each step of the micro-cluster maintenance procedure produce insertions and deletions into the PL list that should affect only a limited subset of the ordering in OL. If we employ incremental techniques such as those proposed in [10], the computational effort to maintain the OL list though time is very small compared to the reward of the mined information, since this depends only on the change to the data structure rather than on the dimensionality or size of the data stream.

In this way the StreamOptics methodology developed here maintains an ordered list OL, of core-micro-clusters, based on their reachability distance. To this end, at each time step, we can use the methodology of OPTICS to produce a reachability plot that depicts the current micro-cluster structure. By keeping track of every such ordering, we can have a record of how the ordering of micro-clusters, and hence the clusters in the data steam, are changing in time.

The reachability plots produced by the OPTICS approach are 2-dimensional. Since we are tracking how these plots change through time, we consider time as being an extra dimension and combine these to form a 3-dimensional plot. Essentially, we combine the reachability "curve" at each time t in order to create a 3-dimensional surface plot, the *reachability surface*, $RS : T \times N \rightarrow R$, where each point (t, i) depicts the reachability distance of the ith micro-cluster in the ordering at time t. In order to better visualise the cluster structure represented by this 3-dimensional surface, we can represent it as a contour plot, to provide further insight into how the clusters are changing in time.

However we should note that the StreamOptics methodology does not guarantee that the position of the micro-clusters remains the same, even when nothing changes in the stream. However, this effect is minimized if the the order of the micro-clusters in PL is re-arranged so that it matches OL. This way when ordering is re-constructed the optical changes should be minimal.

4 Experimental Results

We now provide some results produced by StreamOptics, for both synthetic and real data sets.

4.1 Spawning Clusters

In this section we use the 2-dimensional synthetic data set, $\text{Dset}_{2\text{d}}$. It initially consists of random points drawn sequentially from a finite mixture of two Gaussian distributions randomly placed in $[100, 200]^2$. At time point 2000, we simulate the spawning of a new cluster by introducing one more Gaussian component into the model. In Fig 3, we illustrate the results of StreamOptics at two different time instances. In the top row, 500 data points are drawn from the Gaussian mixture at times 2000, 2100 and 3000, from left to right, respectively. Clearly at time 2100 the points instantiated from the newly introduced Gaussian component start to have an apparent effect in the mixture. In the second row of the figure, the reachability surfaces are exhibited, for the time ranges $[0, 2000]$, and $[2000, 4000]$, from left to right respectively. Finally in the 3rd row the latter surfaces are exhibited as contour plots. As illustrated, the reachability surface attains a constant two valey form as long as the clustering structure remains steady. This is the expected behaviour since nothing is changing in the data stream. Interestingly, when the new component is introduced, this change affects the surface in both the *time* and *ordering* dimensions, since new micro-clusters are introduced to capture the appearance of the new cluster. Gradually a new peak is introduced in the reachability surface, that demonstrates the birth of the new component in the mixture. Finally the surface stabilises again when the new structure of the data is described adequately. There is some case where the surface seems to mess the ordering however this does not seem to affect the acquired knowledge.

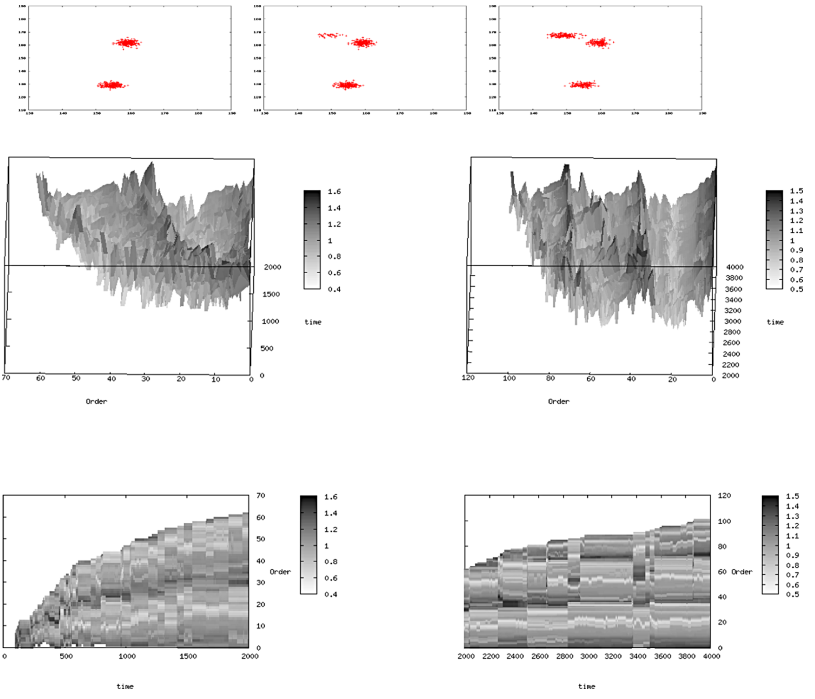


Fig. 3. Dset_{2d} along with the reachability surface produced by StreamOptics for different time instances

4.2 Disappearing Clusters

In this section we apply the algorithm to a case in which clusters disappear. The simulated data set, Dset_{5d}, consists of 5-dimensional vectors, initially drawn from a finite mixture of 10 Gaussian distributions randomly positioned in a closed subset of $\mathbb{R}^5$. From time step 3000, to time step 4000, 7 of the clusters disappear, gradually every 100 time steps. In Fig. 4, we display a contour plot of the reachability surface, as in this case it is more informative. The plot shows that micro-clusters start to disappear. However the surface finally stabilizes, resulting in a stable plot after time 4500.

4.3 The Forest CoverType Data

To examine the algorithm's capabilities with real-world data we employ the Forest CoverType real world data set, obtained from the UCI machine learning repository [11]. This data set, $\text{Dset}_{\text{Forest}}$, is comprised of 581012 observations of 54 attributes, where each observation is labeled as one of seven forest cover classes. We retain the 10 numerical attributes. In Fig. 5, the reachability surface is displayed, between time steps 2000 and 2200. Note that there is a peak in the plot around the 34th-36th micro-cluster in Fig. 5(a). To examine the class

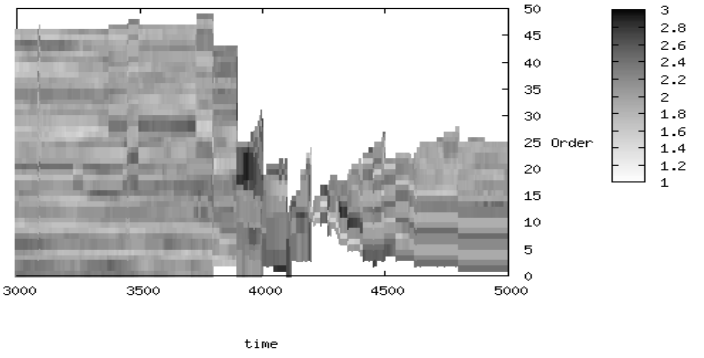
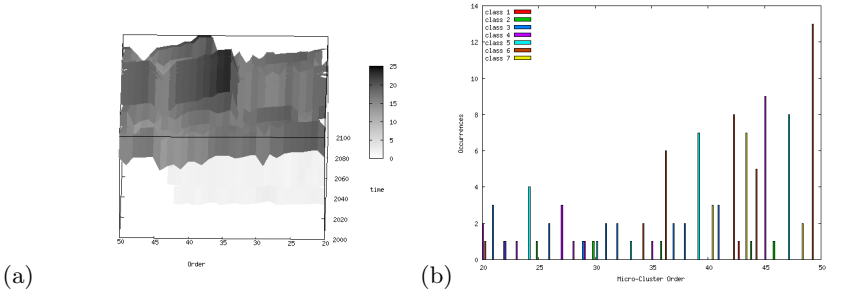


Fig. 4. The contour plot of the reachability surface produced by StreamOptics for Dset_{5d}



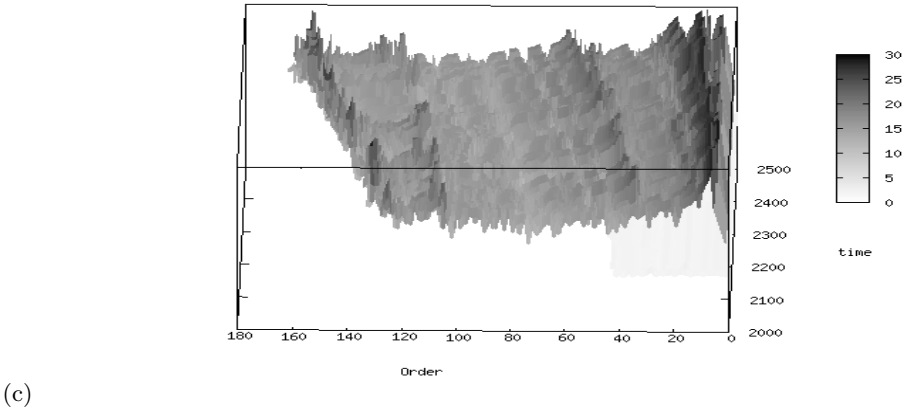


Fig. 5. The results on $\text{Dset}_{\text{Forest}}$

correspondence, in Fig. 5(b), the class of each point in the stream between time steps 2000 and 2200, is plotted against their nearest micro-cluster in the ordering, using a histogram. The latter figure shows that the correspondence of the classes changes at the 34th-36th micro-cluster as was anticipated by the peak present in the same position of the reachability surface. To obtain a wider view of how this structure evolves over time, in Fig 5(c) the reachability surface between times 2000 and 2500 is shown. Clearly, the data set has a somewhat stable clustering, mostly embodied in 3 different clusters, that is persistent through time.

5 Concluding Remarks

Clustering, as one of the most fundamental procedures for extracting information from data, plays a key role in the understanding of massive data streams. Clustering methods have recently been extended [2,5,13], to address some of the problems that emerge with streaming data. However, methods that can visualise the change of the clustering structure through time have only been investigated in lower dimensional situations or via projection [1].

In this work, we hybridise a stream clustering framework with an extension of OPTICS, a successful technique for the visualisation of static clustering [3]. The resulting method is shown through experimental investigation to provide insight into both the clustering structure and its evolution in time. The plots produced by our OpticsStream algorithm allow the user to identify the change in cluster structure in the case of both emerging and fading clusters. The results are also evaluated in a real world setting, where a-priori determined class information is used as a validation measure.

Acknowledgements

This research was undertaken as part of the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Systems) project and is jointly funded by a BAE Systems and EPSRC (Engineering and Physical Research Council) strategic partnership, under EPSRC grant EP/C548051/1.

References

1. Aggarwal, C.: A framework for diagnosing changes in evolving data streams. In: ACM SIGMOD Conference, ACM Press, New York (2003)
2. Aggarwal, C., Han, J., Wang, J., Yu, P.: A framework for projected clustering high dimensional data streams. In: 10th VLDB conference (2004)
3. Ankerst, M., Breunig, M.M., Kriegel, H.-P., Sander, J.: OPTICS: Ordering points to identify the clustering structure. In: Proceedings of ACM-SIGMOD International Conference on Management of Data (1999)
4. Berthold, M., Hand, D.J.: Intelligent Data Analysis: an introduction. Springer, Heidelberg (2003)

5. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: 2006 SIAM Conference on Data Mining (2006)
6. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Machine Learning 42(1/2), 143–175 (2001)
7. Hand, D.J., Heard, N.A.: Finding groups in gene expression data. J Biomed Biotechnol. 2, 215–225 (2005)
8. Hinneburg, A., Keim, D.A., Wawryniuk, M.: HD-Eye: visual mining of high-dimensional data. Computer Graphics and Applications, IEEE 19(5), 22–31 (1999)
9. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys 31(3), 264–323 (1999)
10. Kriegel, H.-P., Kroger, P., Gotlibovich, I.: Incremental OPTICS: Efficient computation of updates in a hierarchical cluster ordering. In: 5th Int. Conf. on Data Warehousing and Knowledge Discovery (2003)
11. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)
12. Sander, J., Ester, M., Kriegel, H.-P., Xu, X.: Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. Data Mining and Knowledge Discovery 2(2), 169–194 (1998)
13. Tasoulis, D.K., Adams, N.M., Hand, D.J.: Unsupervised clustering in streaming data. In: IEEE International Conference on Data Mining (2006)
14. Tryon, C.: Cluster Analysis. Edward Brothers, Ann Arbor, MI (1939)

Relational Topographic Maps

Alexander Hasenfuss and Barbara Hammer

Clausthal University of Technology – Department of Informatics
Clausthal-Zellerfeld, Germany
{hasenfuss,hammer}@in.tu-clausthal.de

Abstract. We introduce relational variants of neural topographic maps including the self-organizing map and neural gas, which allow clustering and visualization of data given as pairwise similarities or dissimilarities with continuous prototype updates. It is assumed that the (dis-)similarity matrix originates from Euclidean distances, however, the underlying embedding of points is unknown. Batch optimization schemes for topographic map formations are formulated in terms of the given (dis-)similarities and convergence is guaranteed, thus providing a way to transfer batch optimization to relational data.

1 Introduction

Topographic maps such as the self-organizing map (SOM) constitute a valuable tool for robust data inspection and data visualization which has been applied in diverse areas such as telecommunication, robotics, bioinformatics, business, etc. [16]. Alternative methods such as neural gas (NG) [20] provide an efficient clustering of data without fixing a prior lattice. This way, subsequent visualization such as multidimensional scaling, e.g. Sammon's mapping [18,26] can readily be applied, whereby no prior restriction of a fixed lattice structure as for SOM is necessary and the risk of topographic errors is minimized. For NG, an optimum (nonregular) data topology is induced such that browsing in a neighborhood becomes directly possible [21].

In the last years, a variety of extensions of these methods has been proposed to deal with more general data structures. This accounts for the fact that more general metrics have to be used for complex data such as microarray data or DNA sequences. Further it might be the case that data are not embedded in a vector space at all, rather, pairwise similarities or dissimilarities are available.

Several extensions of classical SOM and NG to more general data have been proposed: a statistical interpretation of SOM as considered in [4,14,28,29] allows to change the generative model to alternative general data models. The resulting approaches are very flexible but also computationally quite demanding, such that proper initialization and metaheuristics (e.g. deterministic annealing) become necessary when optimizing statistical models. For specific data structures such as time series or recursive structures, recursive models have been proposed as reviewed e.g. in the article [9]. However, these models are restricted to recursive data structures with Euclidean constituents. Online variants of SOM and NG

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 93–105, 2007.

have been extended to general kernels e.g. in the approaches presented in [24,31] such that the processing of nonlinearly preprocessed data becomes available. However, these versions have been derived for kernels, i.e. similarities and (slow) online adaptation only.

The approach [17] provides a fairly general method for large scale application of SOM to nonvectorial data: it is assumed that pairwise similarities of data points are available. Then the batch optimization scheme of SOM can be generalized by means of the generalized median to a visualization tool for general similarity data. Thereby, prototype locations are restricted to data points. This method has been extended to NG in [2] together with a general proof of the convergence of median versions of clustering. Further developments concern the efficiency of the computation [1] and the integration of prior information if available to achieve meaningful visualization and clustering [5,6,30].

Median clustering has the benefit that it builds directly on the derivation of SOM and NG from a cost function. Thus, the resulting algorithms share the simplicity of batch NG and SOM, its mathematical background and convergence. However, for median versions, prototype locations are restricted to the set of given training data which constitutes a severe restriction in particular for small data sets. Therefore, extensions which allow a smooth adaptation of prototypes have been proposed e.g. in [7]. In this approach, a weighting scheme is introduced for the points which represent virtual prototype locations thus allowing a smooth interpolation between the discrete training data. This model has the drawback that it is not an extension of the standard Euclidean version and it gives different results when applied to Euclidean data in a real-vector space.

Here, we use an alternative way to extend NG to relational data given by pairwise similarities or dissimilarities, respectively, which is similar to the relational dual of fuzzy clustering as derived in [12,13]. For a given Euclidean distance matrix or Gram matrix, it is possible to derive the relational dual of topographic map formation which expresses the relevant quantities in terms of the given matrix and which leads to a learning scheme similar to standard batch optimization. This scheme provides identical results as the standard Euclidean version if an embedding of the given data points is known. In particular, it possesses the same convergence properties as the standard variants, thereby restricting the computation to known quantities which do not rely on an explicit embedding in the Euclidean space.

In this contribution, we first introduce batch learning algorithms for standard clustering and topographic map formation derived from a cost function: k-means, neural gas, and the self-organizing map for general (e.g. rectangular, hexagonal, or hyperbolic) grid structures. Then we derive the respective relational dual resulting in a dual cost function and batch optimization schemes for the case of a given distance matrix of data or a given Gram matrix, respectively.

2 Topographic Maps

Neural clustering and topographic maps constitute effective methods for data preprocessing and visualization. Classical variants deal with vectorial data $\boldsymbol{x} \in \mathbb{R}^n$

which are distributed according to an underlying distribution P in the Euclidean plane. The goal of neural clustering algorithms is to distribute prototypes $\boldsymbol{w}^i \in \mathbb{R}^n$, $i = 1, \ldots, k$ among the data such that they represent the data as accurately as possible. A new data point $\boldsymbol{x}$ is assigned to the *winner* $\boldsymbol{w}^{I(\boldsymbol{x})}$ which is the prototype with smallest distance $\|\boldsymbol{w}^{I(\boldsymbol{x})} - \boldsymbol{x}\|^2$. This clusters the data space into the receptive fields of the prototypes.

Different popular variants of neural clustering have been proposed to learn prototype locations from given training data [16]. The well known k-means constitutes one of the most popular clustering algorithms for vectorial data and can be used as a preprocessing step for data mining and data visualization. However, it is quite sensitive to initialization. Unlike k-means, neural gas (NG) [20] incorporates the neighborhood of a neuron for adaptation. Assume the number of prototypes is fixed to k. The cost function is given by

$$E_{\mathrm{NG}}(\boldsymbol{w}) = \frac{1}{2C(\lambda)} \sum_{i=1}^{k} \int h_\lambda(k_i(\boldsymbol{x})) \cdot \|\boldsymbol{x} - \boldsymbol{w}^i\|^2 \, P(d\boldsymbol{x})$$

where

$$k_i(\boldsymbol{x}) = |\{\boldsymbol{w}^j \mid \|\boldsymbol{x} - \boldsymbol{w}^j\|^2 < \|\boldsymbol{x} - \boldsymbol{w}^i\|^2\}|$$

is the rank of the prototypes sorted according to the distances, $h_\lambda(t) = \exp(-t/\lambda)$ scales the neighborhood cooperation with neighborhood range $\lambda > 0$, and $C(\lambda)$ is the constant $\sum_{i=1}^{k} h_\lambda(k_i(\boldsymbol{x}))$. The neighborhood cooperation smoothes the data adaptation such that, on the one hand, sensitivity to initialization can be prevented, on the other hand, a data optimum topological ordering of prototypes is induced by linking the respective two best matching units for a given data point [21]. Classical NG is optimized in an online mode. For a fixed training set, an alternative fast batch optimization scheme is offered by the following algorithm, which in turn computes ranks, which are treated as hidden variables of the cost function, and optimum prototype locations [2]:

init $\boldsymbol{w}^i$
repeat
  compute ranks $k_i(\boldsymbol{x}^j) = |\{\boldsymbol{w}^k \mid \|\boldsymbol{x}^j - \boldsymbol{w}^k\|^2 < \|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2\}|$
  compute new prototype locations $\boldsymbol{w}^i = \sum_j h_\lambda(k_i(\boldsymbol{x}^j)) \cdot \boldsymbol{x}^j / \sum_j h_\lambda(k_i(\boldsymbol{x}^j))$

Like k-means, NG can be used as a preprocessing step for data mining and visualization, followed e.g. by subsequent projection methods such as multidimensional scaling.

The self-organizing map (SOM) as proposed by Kohonen uses a fixed (low-dimensional and regular) lattice structure which determines the neighborhood cooperation. This restriction can induce topological mismatches if the data topology does not match the prior lattice [16]. However, since usually a two-dimensional regular lattice is chosen, this has the benefit that, apart from clustering, a direct visualization of the data results by a representation of the data in the regular lattice space. Thus SOM constitutes a direct data inspection and

visualization method. SOM itself does not possess a cost function, but a slight variation thereof does, as proposed by Heskes [14]. The cost function is

$$E_{\mathrm{SOM}}(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{k}\int \delta_{i,I^*(\boldsymbol{x})} \cdot \sum_k h_\lambda(n(i,k))\|\boldsymbol{x}-\boldsymbol{w}^k\|^2\, P(d\boldsymbol{x})$$

where $n(i,j)$ denotes the neighborhood structure induced by the lattice and $h_\lambda(t) = \exp(-t/\lambda)$ scales the neighborhood degree by a Gaussian function. Thereby, the index $I^*(\boldsymbol{x})$ refers to a slightly altered winner notation: the neuron $I^*(\boldsymbol{x})$ becomes winner for $\boldsymbol{x}$ for which the average distance

$$\sum_k h_\lambda(n(I^*(\boldsymbol{x}),k))\|\boldsymbol{x}-\boldsymbol{w}^k\|^2$$

is minimum. Often, neurons are arranged in a graph structure which defines the topology, e.g. a rectangular or hexagonal tesselation of the Euclidean plane resp. a hyperbolic grid on the two-dimensional hyperbolic plane, the latter allowing a very dense connection of prototypes with exponentially increasing number of neighbors. In these cases, the function $n(i,j)$ often denotes the length of a path connecting the prototypes number i and j in the lattice structure. Original SOM is optimized in an online fashion. As beforehand, for fixed training data, batch optimization is possible by subsequently optimizing assignments and prototype locations.

It has been shown in e.g. [2] that batch optimization schemes of these clustering algorithms converge in a finite number of steps towards a (local) optimum of the cost function, provided the data points are not located at borders of receptive fields of the final prototype locations. In the latter case, convergence can still be guaranteed but the final solution can lie at the border of basins of attraction.

3 Relational Data

3.1 Median Clustering

Relational data x^i are not embedded in a Euclidean vector space, rather, pairwise similarities or dissimilarities are available. Batch optimization can be transferred to such situations using the so-called generalized median [2,17]. Assume, distance information $d(x^i, x^j)$ is available for every pair of data points $x^1, \ldots, x^m$. Median clustering reduces prototype locations to data locations, i.e. adaptation of prototypes is not continuous but takes place within the space $\{x^1, \ldots, x^m\}$ given by the data. We write w^i to indicate that the prototypes need no longer be vectorial. For this restriction, the same cost functions as beforehand can be defined whereby the Euclidean distance $\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2$ is substituted by $d(x^j, w^i) = d(x^j, x^{l_i})$ whereby $w^i = x^{l_i}$. Median clustering substitutes the assignment of $\boldsymbol{w}^i$ as (weighted) center of gravity of data points by an extensive search, setting w^i to the data points which optimize the respective cost function for fixed

assignments. This procedure has been tested e.g. in [2,5]. It has the drawback that prototypes have only few degrees of freedom if the training set is small. Thus, median clustering usually gives inferior results compared to the classical Euclidean versions when applied in a Euclidean setting.

3.2 Training Algorithm

Here we introduce relational clustering for data characterized by pairwise similarities or dissimilarities, whereby this setting constitutes a direct transfer of the standard Euclidean training algorithm to more general settings allowing smooth updates of the solutions. The essential observation consists in a transformation of the cost functions as defined above to their so-called relational dual.

Assume training data $x^1, \ldots, x^m$ are given in terms of pairwise distances $d_{ij} = d(x^i, x^j)^2$. We assume that it originates from a Euclidean distance measure in a possibly high dimensional embedding, that means, we are able to find (possibly high dimensional) Euclidean points $\boldsymbol{x}^i$ such that $d_{ij} = \|\boldsymbol{x}^i - \boldsymbol{x}^j\|^2$. Note that this notation includes a possibly nonlinear mapping (feature map) $x^i \mapsto \boldsymbol{x}^i$ corresponding to the embedding in a Euclidean space, which is not known, such that we cannot directly optimize the above cost functions. The key observation is based on the fact that optimum prototype locations $\boldsymbol{w}^j$ for k-means and batch NG can be expressed as linear combination of data points. Therefore, the unknown distances $\|x^j - w^i\|^2$ can be expressed in terms of known values d_{ij}.

More precisely, assume there exist points $\boldsymbol{x}^j$ such that $d_{ij} = \|\boldsymbol{x}^i - \boldsymbol{x}^j\|^2$. Assume the prototypes can be expressed in terms of data points $\boldsymbol{w}^i = \sum_j \alpha_{ij} \boldsymbol{x}^j$ where $\sum_j \alpha_{ij} = 1$ (as is the case for NG, SOM, and k-means). Then

$$\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2 = (D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i \quad (*)$$

where $D = (d_{ij})_{ij}$ constitutes the distance matrix and $\alpha_i = (\alpha_{ij})_j$ the coefficients. This fact can be shown as follows: assume $\boldsymbol{w}^i = \sum_j \alpha_{ij} \boldsymbol{x}^j$, then

$$\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2 = \|\boldsymbol{x}^j\|^2 - 2 \sum_l \alpha_{il} (\boldsymbol{x}^j)^t \boldsymbol{x}^l + \sum_{l,l'} \alpha_{il} \alpha_{il'} (\boldsymbol{x}^l)^t \boldsymbol{x}^{l'} .$$

On the other hand,

$$\begin{aligned}
&(D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i \\
&= \textstyle\sum_l \|\boldsymbol{x}^j - \boldsymbol{x}^l\|^2 \cdot \alpha_{il} - 1/2 \cdot \sum_{ll'} \alpha_{il} \|\boldsymbol{x}^l - \boldsymbol{x}^{l'}\|^2 \alpha_{il'} \\
&= \textstyle\sum_l \|\boldsymbol{x}^j\|^2 \alpha_{il} - 2 \cdot \sum_l \alpha_{il} (\boldsymbol{x}^j)^t \boldsymbol{x}^l + \sum_l \alpha_{il} \|\boldsymbol{x}^l\|^2 - \sum_{ll'} \alpha_{il'} \alpha_{il'} \|\boldsymbol{x}^l\|^2 \\
&\qquad + \textstyle\sum_{ll'} \alpha_{il} \alpha_{il'} (\boldsymbol{x}^l)^t \boldsymbol{x}^{l'} \\
&= \|\boldsymbol{x}^j\|^2 - 2 \textstyle\sum_l \alpha_{il} (\boldsymbol{x}^j)^t \boldsymbol{x}^l + \sum_{l,l'} \alpha_{il} \alpha_{il'} (\boldsymbol{x}^l)^t \boldsymbol{x}^{l'}
\end{aligned}$$

because of $\sum_j \alpha_{ij} = 1$.

Because of this fact, we can substitute all terms $\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2$ using the equation $(*)$ in batch optimization schemes provided optimum prototypes can be

expressed in terms of data points as specified above. For optimum solutions of NG, k-means, and SOM, it holds $\boldsymbol{w}^i = \sum_j \alpha_{ij} \boldsymbol{x}^j$ whereby

1. $\alpha_{ij} = \delta_{i,I(\boldsymbol{x}^j)} / \sum_j \delta_{i,I(\boldsymbol{x}^j)}$ for k-means,
2. $\alpha_{ij} = h_\lambda(k_i(\boldsymbol{x}^j)) / \sum_j h_\lambda(k_i(\boldsymbol{x}^j))$ for NG, and
3. $\alpha_{ij} = h_\lambda(n(I^*(\boldsymbol{x}^j), i)) / \sum_j h_\lambda(n(I^*(\boldsymbol{x}^j), i)$ for SOM.

This allows to reformulate the batch optimization in terms of relational data using $(*)$. We obtain the algorithm

init α_{ij} with $\sum_j \alpha_{ij} = 1$
repeat
  compute the distance $\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2 = (D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$
  compute optimum assignments based on this distance matrix
    $\tilde{\alpha}_{ij} = \delta_{i,I(\boldsymbol{x}^j)}$ (for k-means)
    $\tilde{\alpha}_{ij} = h_\lambda(k_i(\boldsymbol{x}^j))$ (for NG)
    $\tilde{\alpha}_{ij} = h_\lambda(n(I^*(\boldsymbol{x}^j), i))$ (for SOM)
  compute $\alpha_{ij} = \tilde{\alpha}_{ij} / \sum_j \tilde{\alpha}_{ij}$.

Hence, prototype locations are computed only indirectly by means of the coefficients α_{ij}. Every prototype is represented by a vector which dimensionality is given by the number of data points. The entry at position l of this vector can be interpreted as the contribution of the data point l to this prototype. Thus, this scheme can be seen as an extension of median clustering towards solutions, where the prototypes are determined by a (virtual) mixture of data points instead of just one point.

3.3 Mapping, Quantization Error, Convergence

For clustering, it is also necessary to assign a new data point x (e.g. a data point from the test set) to classes given pairwise distances of the point to the training data $d_j = d(x, x^j)^2$ corresponding to the distance of x from x^j. As before, we assume that this stems from a euclidean metric, i.e. we can isometrically embed x in Euclidean space as $\boldsymbol{x}$ with $d(x, x^j)^2 = \|\boldsymbol{x} - \boldsymbol{x}^j\|^2$. Then the winner can be determined by using the equality

$$\|\boldsymbol{x} - \boldsymbol{w}^i\|^2 = (D(x)^t \cdot \alpha_i) - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$$

where $D(x)$ denotes the vector of distances $D(x) = (d_j)_j = (d(x, x^j)^2)_j$. This holds because of

$$\|\boldsymbol{x} - \boldsymbol{w}^i\|^2 = \|\boldsymbol{x}\|^2 - 2 \sum_l \alpha_{il} \boldsymbol{x}^t \boldsymbol{x}^j + \sum_{ll'} \alpha_{il} \alpha_{il'} (\boldsymbol{x}^l)^t \boldsymbol{x}^{l'}$$

and

$$\begin{aligned} & (D(x)^t \cdot \alpha_i) - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i \\ = & \textstyle\sum_l \alpha_{il} \|\boldsymbol{x} - \boldsymbol{x}^l\|^2 - 1/2 \sum_{ll'} \alpha_{il} \alpha_{il'} \|\boldsymbol{x}^l - \boldsymbol{x}^{l'}\|^2 \\ = & (D(x)^t \cdot \alpha_i) - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i, \end{aligned}$$

because of $\sum_l \alpha_{il} = 1$.

Note that also the quantization error can be expressed in terms of the given values d_{ij} by substituting $\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2$ by $(D\cdot\alpha_i)_j - 1/2\cdot\alpha_i^t\cdot D\cdot\alpha_i$. Thus, relational clustering can be evaluated by the quantization error as usual.

Relational learning gives exactly the same results as standard batch optimization provided the given relations stem from a euclidean metric. Hence, convergence is guaranteed in this case since it holds for the standard batch versions. If the given distance matrix does not stem from a euclidean metric, this equality does no longer hold and the terms $(D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$ can become negative. In this case, one can correct the distance matrix by the γ-spread transform $D_\gamma = D + \gamma(\mathbf{1} - \mathbf{I})$ for sufficiently large γ where $\mathbf{1}$ equals 1 for each entry and $\mathbf{I}$ is the identity.

3.4 Kernels

This derivation allows to use standard clustering agorithms for euclidean dissimilarity data even if the embedding is not known. A dual situation is present if data are characterized by similarities rather than similarities, i.e. the Gram matrix of a kernel or dot product. Since every positive definite kernel induces a euclidean metric, this setting is included in the one described above (the converse is not valid, compare e.g. [27]). A simpler derivation (with the same results) can be obtained substituting distances

$$\|\boldsymbol{x}^j - \boldsymbol{w}^i\|^2 = \|\boldsymbol{x}^j - \sum_l \alpha_{il}\boldsymbol{x}^l\|^2 = (\boldsymbol{x}^j)^t\boldsymbol{x}^j - 2\sum_l \alpha_{il}(\boldsymbol{x}^j)^t\boldsymbol{x}^l + \sum_{l,l'} \alpha_{il}\alpha_{il'}(\boldsymbol{x}^l)^t\boldsymbol{x}^{l'} \; .$$

This directly leads to a kernelized version of batch clustering, see e.g. [11].

3.5 Complexity

Unlike standard Euclidean clustering, relational clustering has time complexity $\mathcal{O}(k \cdot m^2)$ for one epoch and space complexity $\mathcal{O}(k \cdot m)$ where k is the number of prototypes and m the number of data points. Hence, as for the discrete median versions, the complexity becomes quadratic with respect to the training data instead of a linear complexity for the Euclidean case. For small data sets (where 'virtual' interpolation of training data is necessary to obtain good prototype locations), this is obviously not critical. For large data sets, approximation schemes should be considered such as a restriction of the non-zero positions of the vectors α_i to the most prominent data points. We are currently investigating this possibility.

4 Experiments

4.1 Artificial Euclidean Benchmark

As stated before, the relational methods yield the same results as standard batch optimization provided the given relations originate from a Euclidean metric.

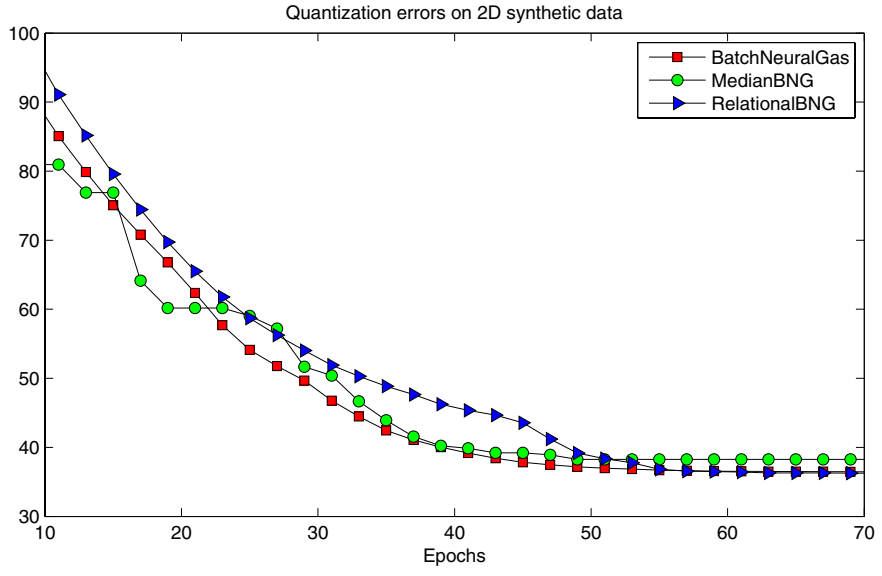


Fig. 1. Quantization errors on the Euclidean Synthetic 2D dataset for different variants of NG. The discussed effects on the performance of the different variants can clearly be observed. Standard Batch (36.42) and the Relational variant (36.27) are virtually on the same level, but the Median version (38.26) is slightly behind due to limited number of locations in discrete data space.

We demonstrate this fact for neural gas on a synthetic dataset taken from [25] consisting of 250 data points. Six neurons were trained for 150 epochs.

The results are depicted in Fig. 1 and 2. It can be clearly observed that the neurons for standard neural gas and relational neural gas are located nearly on the same spots, whereas the median neurons are slightly off due to the limited number of potential locations in the discrete data space. The quantization error is also virtually on the same minimum level for the Euclidean and relational variant, however, the median version is slightly behind again.

4.2 Classification of Protein Families

K-means, neural gas, and SOM, as well as their relational variants presented in this article, perform unsupervised clustering. However, in practice they are often applied to classification tasks. Certainly, the performance of unsupervised prototype-based methods for that kind of task is strongly dependent on the class structures of the data space. Nevertheless, it seems that many real-world datasets are good natured in this respect.

Here, we present results for a classification task in bioinformatics. The data is given by the evolutionary distance of 226 globin proteins which is determined

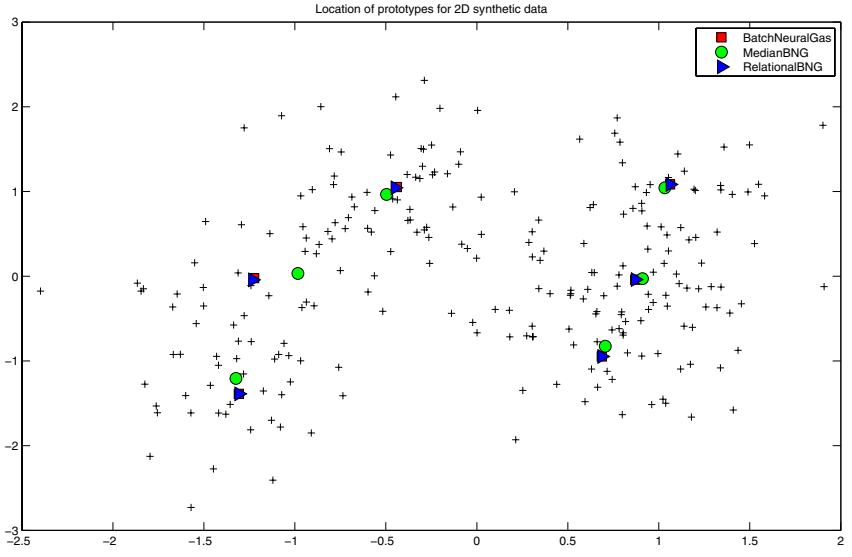


Fig. 2. Prototype locations on the Euclidean Synthetic 2D dataset for different variants of NG. The discussed effects on the location of the prototypes can clearly be observed. Standard Batch and the Relational variant are nearly on the same spots, but the Median version is slightly off due to limited number of locations in discrete data space.

by alignment as described in [22]. These samples originate from different protein families: hemoglobin-α, hemoglobin-β, myoglobin, etc. Here, we distinguish five classes as proposed in [10]: HA, HB, MY, GG/GP, and others. For training we use 45 neurons and 150 epochs per run. The results reported in Table 1 are gained from repeated 10-fold stratified cross-validation averaged over 100 repetitions. For comparison, a (supervised) 1-nearest neighbor classifier yields an accuracy 91.6 for our setting (k-nearest neighbor for larger k is worse; [10]).

The advantage of the relational variants with continuous prototype updates for (dis-)similarity data can be observed immediately in terms of better accuracy.

4.3 Topographic Mapping of Protein Families

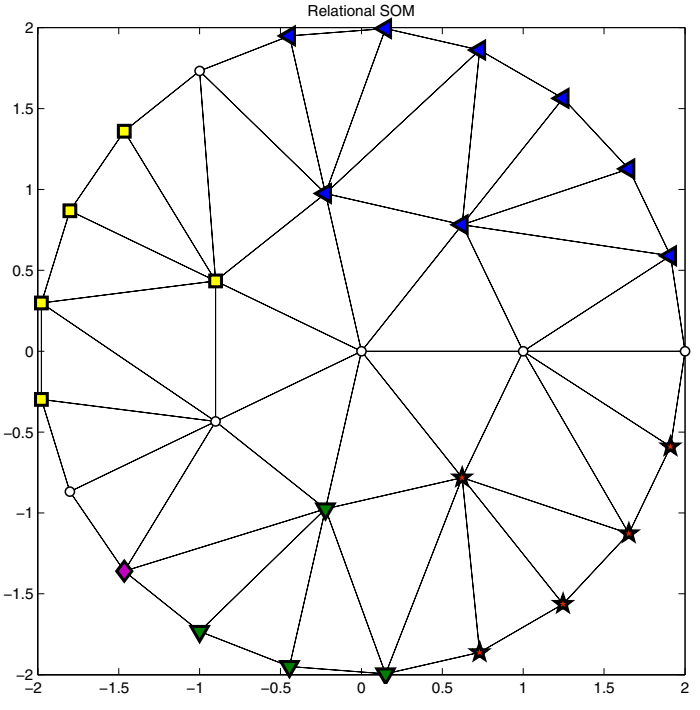
The Protein dataset as described above is mapped by a Relational SOM with 29 neurons and a hyperbolic grid structure. Figure 3 shows the grid projection to the Euclidean plane, the neurons are labeled with class information determined by a majority vote on their receptive fields.

Note that the different clusters can easily be identified. The Relational SOM provides an improved technique to explore dissimilarity data, revealing the structures of interest.

Table 1. Classification accuracy on the Protein Data Set and the Copenhagen Chromosomes dataset, respectively, for posterior labeling

	Median k-Means	Median Batch NG	Relational k-Means	Relational Batch NG
Accuracy (Proteins)				
Mean	76.1	76.3	88.0	89.9
StdDev	1.3	1.8	1.8	1.3
Accuracy (Chromosomes)				
Mean	82.3	82.8	90.6	91.3
StdDev	2.2	1.7	0.6	0.2

Fig. 3. Mapping of the non-Euclidean Protein dataset by a Relational SOM with hyperbolic grid structure



4.4 Chromosome Images

The Copenhagen chromosomes database is a benchmark from cytogenetics [19]. A set of 4200 human nuclear chromosomes from 22 classes (the X resp. Y sex chromosome is not considered) are represented by the grey levels of their images and transferred to strings representing the profile of the chromosome by the thickness

of their silhouettes. Thus, this data set consists of strings of different length. The edit distance is a typical distance measure for two strings of different length, as described in [15,23]. In our application, distances of two strings are computed using the standard edit distance whereby substitution costs are given by the signed difference of the entries and insertion/deletion cost are given by 4.5 [23].

The algorithms were tested in a repeated 2-fold cross-validation using 100 neurons and 100 epochs per run. The results presented are the mean accuracy over 10 repeats of the cross-validation. Results are reported in Tab. 1. As can be seen, relational clustering achieves an accuracy of more than 90% which is an improvement by more 8% compared to median variants. This is comparable to the classification accuracy of hidden Markov models as reported in [15].

5 Discussion

We have introduced relational neural clustering which extends the classical Euclidean versions to settings where pairwise Euclidean distances (or similarities) of the data are given but no explicit embedding into a Euclidean space is known. By means of the relational dual, batch optimization can be formulated in terms of these quantities only. This extends previous median clustering variants to a continuous prototype update which is particularly useful for only sparsely sampled data.

The general framework as introduced in this article opens the way towards the transfer of further principles of SOM and NG to the setting of relational data: as an example, the magnification factor of topographic map formation for relational data transfers from the Euclidean space, and possibilities to control this factor as demonstrated for batch clustering e.g. in the approach [8] can readily be used.

Since these relational variants rely on the same cost function as the standard Euclidean batch optimization schemes, extensions to additional label information as proposed for the standard variants [5,6] become available.

One very important subject of future work concerns the complexity of computation and sparseness of prototype representation. For the approach as introduced above, the complexity scales quadratic with the number of training examples. For SOM, it would be worthwhile to investigate whether efficient alternative computation schemes such as proposed in the approach [1] can be derived. Furthermore, the representation contains a large number of very small coefficients, which correspond to data points for which the distance from the prototype is large. Therefore it can be expected that a restriction of the representation to the close neighborhood is sufficient for accurate results.

References

1. Conan-Guez, B., Rossi, F., El Golli, A.: A fast algorithm for the self-organizing map on dissimilarity data. In: Workshop on Self-Organizing Maps, pp. 561–568 (2005)
2. Cottrell, M., Hammer, B., Hasenfuss, A., Villmann, T.: Batch and median neural gas. Neural Networks 19, 762–771 (2006)

3. Graepel, T., Herbrich, R., Bollmann-Sdorra, P., Obermayer, K.: Classification on pairwise proximity data. In: Jordan, M.I., Kearns, M.J., Solla, S.A. (eds.) NIPS, vol. 11, pp. 438–444. MIT Press, Cambridge (1999)
4. Graepel, T., Obermayer, K.: A stochastic self-organizing map for proximity data. Neural Computation 11, 139–155 (1999)
5. Hammer, B., Hasenfuss, A., Schleif, F.-M., Villmann, T.: Supervised median neural gas. In: Dagli, C., Buczak, A., Enke, D., Embrechts, A., Ersoy, O. (eds.) Intelligent Engineering Systems Through Artificial Neural Networks 16. Smart Engineering System Design, pp. 623–633. ASME Press (2006)
6. Hammer, B., Hasenfuss, A., Schleif, F.-M., Villmann, T.: Supervised batch neural gas. In: Schwenker, F. (ed.) Proceedings of Conference Artificial Neural Networks in Pattern Recognition (ANNPR), pp. 33–45. Springer, Heidelberg (2006)
7. Hasenfuss, A., Hammer, B., Schleif, F.-M., Villmann, T.: Neural gas clustering for dissimilarity data with continuous prototypes. In: Sandoval, F., Prieto, A., Cabestany, J., Graña, M. (eds.) IWANN 2007. LNCS, vol. 4507, Springer, Heidelberg (2007)
8. Hammer, B., Hasenfuss, A., Villmann, T.: Magnification control for batch neural gas. Neurocomputing 70, 1225–1234 (2007)
9. Hammer, B., Micheli, A., Sperduti, A., Strickert, M.: Recursive self-organizing network models. Neural Networks 17(8-9), 1061–1086 (2004)
10. Haasdonk, B., Bahlmann, C.: Learning with distance substitution kernels. In: Rasmussen, C.E., Bülthoff, H.H., Schölkopf, B., Giese, M.A. (eds.) Pattern Recognition. LNCS, vol. 3175, Springer, Heidelberg (2004)
11. Hammer, B., Hasenfuss, A.: Relational Topographic Maps, Technical Report IfI-07-01, Clausthal University of Technology, Institute of Informatics (April 2007)
12. Hathaway, R.J., Bezdek, J.C.: Nerf c-means: Non-euclidean relational fuzzy clustering. Pattern Recognition 27(3), 429–437 (1994)
13. Hathaway, R.J., Davenport, J.W., Bezdek, J.C.: Relational duals of the c-means algorithms. Pattern Recognition 22, 205–212 (1989)
14. Heskes, T.: Self-organizing maps, vector quantization, and mixture modeling. IEEE Transactions on Neural Networks 12, 1299–1305 (2001)
15. Juan, A., Vidal, E.: On the use of normalized edit distances and an efficient k-NN search technique (k-AESA) for fast and accurate string classification. In: ICPR 2000, vol. 2, pp. 680–683 (2000)
16. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
17. Kohonen, T., Somervuo, P.: How to make large self-organizing maps for nonvectorial data. Neural Networks 15, 945–952 (2002)
18. Kruskal, J.B.: Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika 29, 1–27 (1964)
19. Lundsteen, C., Phillip, J., Granum, E.: Quantitative analysis of 6985 digitized trypsin G-banded human metaphase chromosomes. Clinical Genetics 18, 355–370 (1980)
20. Martinetz, T., Berkovich, S.G., Schulten, K.J.: Neural-gas' network for vector quantization and its application to time-series prediction. IEEE Transactions on Neural Networks 4, 558–569 (1993)
21. Martinetz, T., Schulten, K.: Topology representing networks. Neural Networks 7, 507–522 (1994)
22. Mevissen, H., Vingron, M.: Quantifying the local reliability of a sequence alignment. Protein Engineering 9, 127–132 (1996)
23. Neuhaus, M., Bunke, H.: Edit distance based kernel functions for structural pattern classification. Pattern Recognition 39(10), 1852–1863 (2006)

24. Qin, A.K., Suganthan, P.N.: Kernel neural gas algorithms with application to cluster analysis. In: ICPR 2004, vol. 4, pp. 617–620 (2004)
25. Ripley, B.D.: Pattern Recognition and Neural Networks, Cambridge (1996)
26. Sammon Jr., J.W.: A nonlinear mapping for data structure analysis. IEEE Transactions on Computers 18, 401–409 (1969)
27. Schölkopf, B.: The kernel trick for distances, Microsoft TR 2000-51 (2000)
28. Seo, S., Obermayer, K.: Self-organizing maps and clustering methods for matrix data. Neural Networks 17, 1211–1230 (2004)
29. Tino, P., Kaban, A., Sun, Y.: A generative probabilistic approach to visualizing sets of symbolic sequences. In: Kohavi, R., Gehrke, J., DuMouchel, W., Ghosh, J. (eds.) Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD-2004, pp. 701–706. ACM Press, New York (2004)
30. Villmann, T., Hammer, B., Schleif, F., Geweniger, T., Herrmann, W.: Fuzzy classification by fuzzy labeled neural gas. Neural Networks 19, 772–779 (2006)
31. Yin, H.: On the equivalence between kernel self-organising maps and self-organising mixture density network. Neural Networks 19(6), 780–784 (2006)

Incremental Learning with Multiple Classifier Systems Using Correction Filters for Classification*

José del Campo-Ávila, Gonzalo Ramos-Jiménez, and Rafael Morales-Bueno

Departamento de Lenguajes y Ciencias de la Computación
E.T.S. Ingeniería Informática. Universidad de Málaga
Málaga, 29071, Spain
{jcampo,ramos,morales}@lcc.uma.es

Abstract. Classification is a quite relevant task within data mining area. This task is not trivial and some difficulties can arise depending on the nature of the problem. Multiple classifier systems have been used to construct ensembles of base classifiers in order to solve or alleviate some of those problems. One of the most current problems that is being studied in recent years is how to learn when the datasets are too large or when new information can arrive at any time. In that case, incremental learning is an approach that can be used. Some works have used multiple classifier system to learn in an incremental way and the results are very promising. The aim of this paper is to propose a method for improving the classification (or prediction) accuracy reached by multiple classifier systems in this context.

1 Introduction

Classification and prediction tasks are two of the most popular activities in data mining and there are many approaches that try to extract knowledge from data. These approaches are very diverse, but one of the most active research area is focused on multiple classifier systems what have been benefited from the idea of using a committee or ensemble of models to do that tasks. In the literature we can find many approaches to define a multiple classifier system, but two of the most representative methods are bagging [1] and boosting [2].

Machine learning systems are currently required to deal with large datasets. But, trying to extract knowledge from such numbers of examples can become an inaccessible problem for traditional algorithms (simple or multiple classifier systems). Moreover, data streams have recently become a new challenge for data mining because of certain features [3] of which infinite data flow is the most notable. Incremental learning methods (including online or sequential methods) have some features that make them very suitable to deal with huge amounts of data and to prevent excessively high memory requirements.

* This work has been partially supported by the FPI program and the MOISES-TA project, number TIN2005-08832-C03, of the MEC, Spain.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 106–117, 2007.

In recent years some researches have tackled the incremental learning methods using multiple classifier systems. They have been made using different base classifiers: decision trees [4,5], neural networks and Naïve Bayes [6], etc. Those approaches are susceptible of being improved independently of the base classifier: they can be trained with rough set based reducts [7,8], they can use weighted majority voting procedures [9], they can specifically deal with concept drift [3,5,10], etc.

In this paper we propose a method to improve the classification (or prediction) accuracy reached by multiple classifier systems. It is also independent on the kind of the ensemble and can be applied to a wide variety of multiple classifier systems.

The next Section describes a basic method for developing incremental learning methods using multiple classifier systems. In that same Section (Subsection 2.2) one new method to improve the accuracy of the model is exposed. It uses what we have called "*correction filters for classification*". The experimental evaluation is presented in Section 3. Finally, in Section 4, we summarise our conclusions and suggest future lines of research.

2 Incremental Learning with Multiple Classifier Systems

The advantages of using multiple classifier systems have been shown in many different works [1,2]. Recently, some research efforts are being directed to use those advantages in the incremental learning area [4,5,6]. Nowadays, this is very necessary because the size of the datasets are increasingly growing. Even more, the data streams are becoming a fact and incremental learning has been revealed as an effective approach to deal with this emergent problem.

In the next Subsection we will present one of the most common approaches for using multiple classifier systems in an incremental way. Then, in Subsection 2.2, we will describe how almost every approach based in multiple classifier systems can be improved using correction filters for classification, the contribution that we propose.

2.1 Basic Method

In general, as proposed Fern and Givan [10], the methods for using multiple classifier systems for incremental learning can be divided in two main groups. We can distinguish between *sequential* or *parallel* generation. The generation is sequential when the classifiers that constitute the ensemble are induced one at a time, ceasing to update each member once the next one is started. On the other hand, the approach is called parallel when every classifier in the ensemble is updated every time that a new example is used. In this Subsection we will present the sequential approach because the first works were based on it [4,11]; anyway, this selection does not condition following sections.

In Figure 1 we represent a simple diagram where the induction of a multiple classifier system for incremental learning is shown. The process used for that

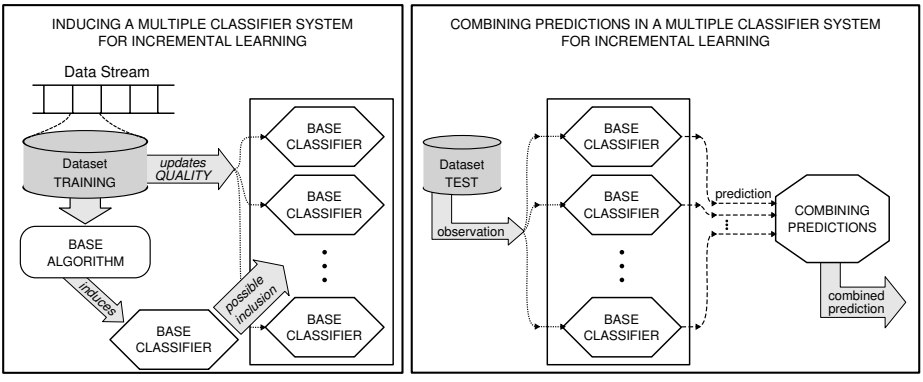


Fig. 1. Simple diagrams describing the induction of a multiple classifier system for incremental learning (left) and its application to the prediction process (right)

induction can be very useful when we have to work with very large datasets (or data streams) because small blocks of examples are taken to induce the base classifiers. By selecting small blocks, different classifier algorithms can be used to induce such base classifiers and we solve the problem of managing large amounts of data. In addition, the blocks are used to update the quality measures of the other base classifiers that constitute the multiple classifier systems. In the same Figure 1 (on the right side) we have shown how the multiple classifier systems are normally used to predict (this procedure is also used by non incremental multiple classifier systems).

The pseudocode of a basic approach for incremental learning with multiple classifier systems is shown in Figure 2. It follows the schema presented in Figure 1 and it is based on the algorithm SEA, proposed by Street and Kim [5]. As can be seen, the dataset (large database or data stream) is used as the source of examples, and they are extracted from it using blocks of equal size. Once a block is composed, it is used to induce a new base classifier that will be added to the ensemble. The quality of every classifier in the ensemble is updated (the models keep equal) and someone with worse quality than the new induced one is removed from it (if a maximum number of base classifiers is exceeded).

We must note that, although the algorithm stops when there are no more examples in the dataset, the model (multiple classifier system) that is being induced can be used at any instant to classify an example or predict the class of one observation. This method produces one kind of learning that can be designated as "*any-time*" learning [12] because, at every moment, there is a multiple classifier system ready to be used (to classify or predict).

We have developed a multiple classifier system for incremental learning based on the pseudocode presented in Figure 2. It is called MultiCIDIM-DS and its name comes from the base algorithm used: CIDIM [13]. This algorithm, CIDIM, has also been used in other multiple classifier systems (like FE-CIDIM [14]) obtaining good results. The main advantage of using CIDIM as base algorithm

Input: *dataset*, *base_algorithm*, *block_size*, *max_number_of_classifiers*

1. $Ensemble = \varnothing$
2. **while** $dataset \neq \varnothing$ **do:**
2.1 $Training\_Examples \leftarrow$ Extract first *block_size* examples from *dataset* (they are removed from *dataset*)
2.2. $Classifier_i \leftarrow$ Induce new classifier with *base_algorithm* using $Training\_Examples$
2.3. Update $Quality(Classifier_{i-1})$ using $Training\_Examples$
2.4. Update $Quality(x)\ \forall x \in Ensemble$ using $Training\_Examples$
2.5. **if** $|Ensemble| < max\_number\_of\_classifiers$ **then:**
2.5.1. Insert $Classifier_{i-1}$ in $Ensemble$
2.6. **else:**
2.6.1. **if** $\exists x \in Ensemble \mid Quality(x) < Quality(Classifier_{i-1})$ **then:**
2.6.1.1. Replace x with $Classifier_{i-1}$ in $Ensemble$

Output: *Ensemble*

Fig. 2. Pseudocode for incremental learning with multiple classifier systems

is that it induces accurate (and small) decision trees. Besides the different base algorithms used by SEA (C4.5 [15]) and MultiCIDIM-DS (CIDIM), there are two main differences between them: the quality measure used by MultiCIDIM-DS is simpler, because it only uses the success rate; and the base classifier selected to be removed, when the size of the ensemble is exceeded, is the worst one (instead of any worse than the new induced one).

2.2 Correction Filters for Classification

Different improvements have been incorporated to multiple classifier systems: using weighted majority voting procedure [9], training with rough set based reducts [7,8], etc. Our objective in this paper is to present a new improvement that can be used in combination with any kind of multiple classifier system for incremental learning. The purpose of the method is to learn which subspaces of the global space are correctly learnt by each base classifier, as Ortega did with non-incremental multiple classifier systems [16]. Considering this knowledge a new more informed voting method can be designed. A "*gated*" voting procedure had been experimented by Street and Kim [5] but they got no consistent improvement over simple voting. The procedure proposed by them consisted in training a separate classifier in conjunction with the base classifier. The new classifier would be trained to learn if the base classifier would correctly classify a concrete example (no more details are given because they got *no consistent improvement*). In fact, the idea is similar to the one that we present in this paper, except that we are considering incremental algorithms and we are achieving some improvements.

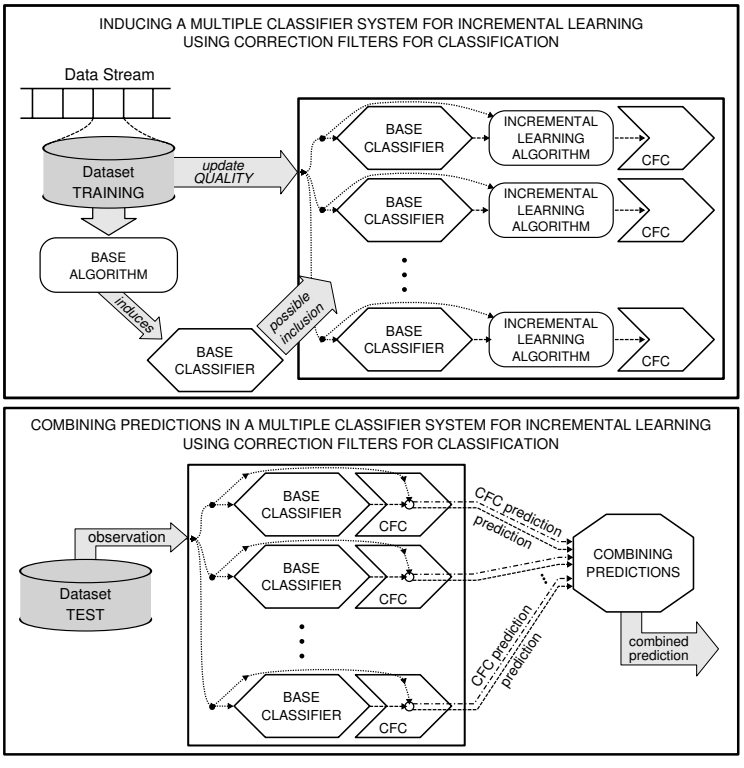


Fig. 3. Simple diagrams describing the induction of a multiple classifier system for incremental learning with correction filters for classification or CFC (top) and its application to the prediction process (bottom)

As we have mentioned, the method that we are proposing includes some classifiers that try to learn which kind of examples are correctly classified by the corresponding base classifiers. In Figure 3 is shown a schematic description. These new classifiers (called *correction filters*) are induced by any kind of incremental algorithm [17,18]. Thus, the ensemble will be constituted by a maximum number of base classifiers and the same number of correction filters for classification (CFC). We have called the new classifiers as correction filters because they will be used to filter the output of the base classifiers when the ensemble is used to predict. We must note that the base classifiers remain unalterable, what change are their quality measures and their correction filters.

The examples passed to the algorithm that induces the correction filters are the same that the examples used to induce new base classifiers (or to update the quality measure of existing base classifiers) but with one difference: the class attribute. The original class attribute is replaced by a binary attribute that represents the success or failure of the corresponding base classifier when it is used to classify the original example. Thus, the correction filter will learn

Input: *dataset*, *base_algorithm*, *block_size*, *max_number_of_classifiers*
incremental_algorithm

1. $Ensemble = \varnothing$
2. **while** $dataset \neq \varnothing$ **do:**
2.1 $Training\_Examples \leftarrow$ Extract first *block_size* examples from *dataset* (they are removed from *dataset*)
2.2. $Classifier_i \leftarrow$ Induce new classifier with *base_algorithm* using $Training\_Examples$
2.3. $Ensemble = Ensemble \cup \{Classifier_{i-1}\}$
2.4. **for** every classifier $x \in Ensemble$ **do:**
2.4.1 Update $Quality(x)$ using $Training\_Examples$
2.4.2 $Training\_CFC \leftarrow$ Modify examples in $Training\_Examples$ including the classification made by x
2.4.3 **if** CFC_x exists **then:**
2.4.3.1 $CFC_x \leftarrow$ Update CFC_x with *incremental_algorithm* using $Training\_CFC$
2.4.4 **else:**
2.4.4.1 $CFC_x \leftarrow$ Induce new correction filter with *incremental_algorithm* using $Training\_CFC$
2.5. **if** $|Ensemble| > max\_number\_of\_classifiers$ **then:**
2.5.1. $Worst\_classifier = \{x \in Ensemble \mid Quality(x) < Quality(y) \; \forall y \in Ensemble \wedge x \neq y\}$
2.5.2. $Ensemble = Ensemble - \{Worst\_classifier\}$

Output: *Ensemble*

Fig. 4. Pseudocode for incremental learning with multiple classifier systems using correction filters for classification (CFC)

which examples are correctly classified by its corresponding base classifier and which are misclassified. In the upper side of Figure 3 we present schematically how are induced the CFC and, in Figure 4, we give a detailed description of such process. We have said that the correction filters can be induced by any incremental algorithm. Actually, they can be induced by any kind of classifier algorithm, but using incremental algorithms has some advantages. Some base classifiers can stay in the ensemble for a long time (may be from the beginning). In fact, if their quality is very high (they are quite accurate, simple, etc.) they may be in the ensemble forever. Taking this into account, it would be very desirable to have as exact information as possible about the subspaces that are correctly learnt by every base classifier. The best way of achieving this aim is considering every example with an incremental approach.

This new method preserves the any-time property of the basic method. The ensemble can be used at any time to classify (or predict) any example. The prediction procedure using the new approach is very simple and it is presented in the lower side of Figure 3. Every base classifier gives its own prediction, but the

Input: *Ensemble* (Multiple Classifier System with CFC),
o (observation = unlabelled example)

1. $N \leftarrow$ number of base classifiers in *Ensemble*
2. $c_i \leftarrow$ prediction vector for o using the i-th base classifier in *Ensemble*
3. $f_i \leftarrow$ probability of correct classification of o using the i-th base classifier in *Ensemble* (it is calculated using the i-th CFC)
4. $Prob\_Best\_Prediction = \max\{f_i \mid 1 \geq i \geq N\}$
5. $Min\_Value\_For\_Combination = \max(0.5, (Prob\_Best\_Prediction - (Prob\_Best\_Prediction * 0.1)))$
6. $Predict\_Vector = (0, 0, \ldots, 0)$
7. **if** $Prob\_Best\_Prediction \geq 0.5$ **then:**
7.1. **from** $i = 1$ **to** N **do:**
7.1.1. **if** $f_i \geq Min\_Value\_For\_Combination$ **then:**
7.1.1.1. $Predict\_Vector = Predict\_Vector + c_i * f_i$
8. **else:**
8.1. **from** $i = 1$ **to** N **do:**
8.1.1. $Predict\_Vector = Predict\_Vector + c_i * (1 - f_i)$
8.2. **from** $j = 1$ **to** $|Vector\_Pred|$ **do:**
8.2.1. $Predict\_Vector_j = 1 \ / \ Predict\_Vector_j$
9. $Predict\_Vector = normalize(Predict\_Vector)$

Output: *Predict_Vector* (prediction vector)

Fig. 5. Pseudocode of prediction process using multiple classifier systems with correction filters for classification (CFC)

ensemble only considers the relevant information. It is at this moment when filters are used to correct the classification proposed by the base classifiers. The example without the class label is given to the correction filters, and they give the probabilities of success of the corresponding base classifiers.A more detailed description is shown in Figure 5. The predictions given by the base classifiers whose probability of success is near to the best one ($f_i \geq Min\_Value\_For\_Combination$) are considered to calculate the final prediction vector. If none filter gives a probability of correct classification greater than 0.5 ($Prob\_Best\_Prediction < 0.5$), it means that every base classifier misclassifies the example. In that case, we look for the minority class.

Using sequential or parallel generation does not condition the use of the method that we are presenting, because the correction filters can be always updated independently of the kind of generation. When new examples are used, they are classified by the current base classifiers, and successes or failures are learnt by the incremental algorithm.

Finally, another interesting point that will be briefly discussed in this paper is the possibility of using this method to improve the performance of a multiple classifier system in presence of concept drift. When sequential generation is used and a concept drift happens, the ensemble will probably need to replace most

of the base classifiers. But this will require as many iterations as base classifiers need to be replaced. On the other hand, if the correction filters are induced using an incremental algorithm that can detect concept drift, every base classifier that will be replaced in subsequent steps can be deleted from the ensemble in the first iteration. In this way, classifiers that are incorrect because of the concept drift are promptly removed.

We have used the correction filters for classification to develop a concrete algorithm that we have called MultiCIDIM-DS-CFC. It is based on MultiCIDIM-DS but the difference is the incorporation of CFC. The *base_algorithm* given in the input is CIDIM and the *incremental_algorith* that induces the CFC is IADEM-2 [19].

3 Experiments and Results

In this Section we show the results of different experiments that have been made to test the improvements of using correction filters for classification. The experiments have been done using three synthetic datasets randomly generated using decision trees and two real datasets: the *LED* dataset from *UCI Machine Learning Repository* [20] and the *Electricity* dataset from `http://www.liacc.up.pt/~jgama/ales/elec.tar`. Every dataset built using randomly generated decision trees has one million examples and they are defined by 20 nominal attributes. They have been called *Syn_1* (0% noise and balanced), *Syn_2* (0% noise and imbalanced) and *Syn_3* (15% noise). The dataset created using the *LED* generator also has one million examples and it is noisy (10% noise). The *Electricity* dataset has 27549 examples after removing examples with unknown values.

We have conducted 10-fold cross validations for every experiment. For studying the performance we have selected two criteria: the classification accuracy and the size of the trees. The results shown are the average value and the standard deviation. We are mainly interested in evaluating the improvement of using CFC, so we will set the algorithm MultiCIDIM-DS-CFC as our reference algorithm. We will compare different algorithms with it and we would like to know when there exist statistical differences between the algorithm that uses CFC and other algorithms ($\oplus$ if the algorithm is significantly better than MultiCIDIM-DS-CFC and $\ominus$ if it is worse). In order to show such differences we have conducted a Wilcoxon signed rank test. The reasons for selecting this non-parametric test are well exposed in the work of Demšar [21], but we must remark that it is safer than parametric tests and stronger than some other tests. The confidence level to conduct the rank test has been set to 0.95.

Both versions of MultiCIDIM-DS (with CFC or without CFC) have been compared with other algorithms. As two traditional ensemble algorithms we have chosen bagging [1] and boosting [2]. Their base classifier is C4.5 [15]. We have used the implementation of bagging, boosting and C4.5 given in Weka [22] (C4.5 is called J48). Bagging-J48 and Boosting-J48 do not use an incremental approach and they use the entire dataset to induce the model, what may affect on a higher accuracy. As an incremental algorithm we have selected VFDT [17].

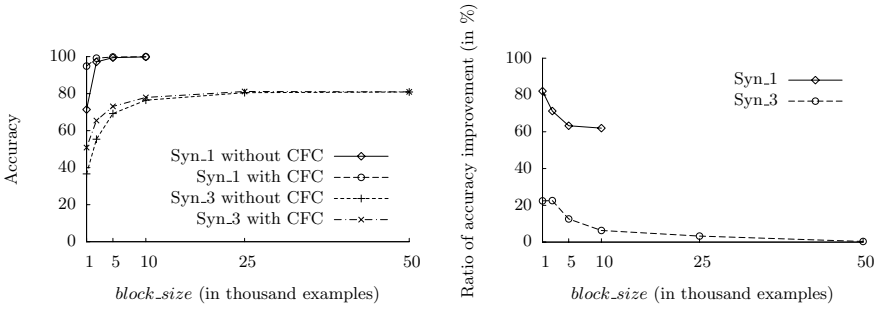


Fig. 6. Accuracy and ratio of accuracy improvement depending on the block size (*block_size*)

We have selected these algorithms because they are well-documented algorithms, they induce decision trees, and their implementations can be found available in the Internet. We must note that all algorithms have been executed using their default configuration.

The only new parameter in MultiCIDIM-DS that is not present in CIDIM, or FE-CIDIM or IADEM-2 is *block_size*, and following we will see its importance. In Figure 6 it can be seen how different values of *block_size* (from 1K to 50K) affects to the accuracy. The default value has been fixed in 10K when there is no noise and 50K when there is noise (one exception has been made with the *Electricity* dataset because its size is not so large and we have used 5K in that case). Another interesting point that can be observed is how using correction filters improves the accuracy (in presence of noise or not). The improvement is greater when the maximum level of accuracy has not been reached yet. Anyway, the improvement is usually significant even when maximum accuracy is reached (as can be seen in Table 1). We have included the ratio of accuracy improvement in Figure 6 because it can not be clearly observed in absolute accuracy chart. As can be seen, although the improvement seems to be small in the *Syn_1* dataset, it is always over 50%, even when accuracy is near to the maximum.

As it can be seen in Table 1, both MultiCIDIM-DS and MultiCIDIM-DS-CFC seems to have a good performance. We must note that, although the results are usually the best ones using Bagging-J48 and Boosting-J48 in experiments without noise (*Syn_1* and *Syn_2*), MultiCIDIM-DS-CFC gets very similar results, overcoming them only in the number of leaves when testing *Syn_1* dataset. On the other hand, we can see how traditional algorithm are not good for dealing with large datasets and the using of an incremental learning approach is revealed as fundamental. Bagging-J48 and Boosting-J48 needed much more memory in order to induce models when the datasets are large and there is noise in them (*Syn-3* and *LED*). Algorithms marked with asterisks ('**') needed 2GB of memory while normal configuration used only a maximum of 512MB. We can see another interesting point when we try to extract knowledge from noisy datasets: the accuracy achieved by Bagging-J48 and Boosting-J48 is usually higher, but

Table 1. Results for the synthetic and real datasets. Algorithms marked with '**' needed 2GB of memory to conclude the induction of the model. We use ⊕ when the result of an algorithm is significantly better than the model induced by MultiCIDIM-DS-CFC and ⊖ when it is worse.

Dataset	Algorithm	Accuracy	Leaves
Syn-1			
0% noise	VFDT	99.55 ± 0.03 ⊖	449.00 ± 0.00 ⊖
	Bagging-J48	100.00 ± 0.00 ⊕	398.00 ± 0.00 ⊖
	Boosting-J48	100.00 ± 0.00 ⊕	398.00 ± 0.00 ⊖
	MultiCIDIM-DS	99.86 ± 0.06 ⊖	345.05 ± 7.38
	MultiCIDIM-DS-CFC	99.947 ± 0.02	345.05 ± 7.38
Syn-2			
0% noise	VFDT	92.68 ± 6.01 ⊖	193.40 ± 35.95
	Bagging-J48	100.00 ± 0.00 ⊕	109.45 ± 6.86 ⊕
	Boosting-J48	100.00 ± 0.00 ⊕	104.50 ± 14.23 ⊕
	MultiCIDIM-DS	99.78 ± 0.36 ⊖	187.49 ± 91.37
	MultiCIDIM-DS-CFC	99.92 ± 0.12	187.49 ± 91.37
Syn-3			
15% noise	VFDT	73.23 ± 5.99 ⊖	1357.20 ± 356.10
	Bagging-J48**	84.14 ± 0.12 ⊕	67626.04 ± 214.63 ⊖
	Boosting-J48**	83.16 ± 0.13 ⊕	134866.69 ± 218.84 ⊖
	MultiCIDIM-DS	80.89 ± 0.99	1354.19 ± 39.25
	MultiCIDIM-DS-CFC	80.96 ± 1.07	1354.19 ± 39.25
LED			
10% noise	VFDT	73.85 ± 0.05 ⊖	75.30 ± 0.95 ⊕
	Bagging-J48**	73.94 ± 0.13	36150.17 ± 197.38 ⊖
	Boosting-J48**	Out of memory	
	MultiCIDIM-DS	73.96 ± 0.22	118.35 ± 5.27
	MultiCIDIM-DS-CFC	73.98 ± 0.14	118.35 ± 5.27
Electricity			
	VFDT	65.08 ± 2.89	29.40 ± 5.97 ⊕
	Bagging-J48	67.27 ± 0.03 ⊕	2689.25 ± 232.16 ⊖
	Boosting-J48	67.90 ± 0.03 ⊕	2347.53 ± 113.99 ⊖
	MultiCIDIM-DS	64.28 ± 3.55 ⊖	320.92 ± 35.73
	MultiCIDIM-DS-CFC	65.57 ± 2.78	320.92 ± 35.73

the average size of the decision trees that constitute the multiple classifier systems is much more higher. Here we can see one advantage of using CIDIM as base classifier: the induction of simpler models.

If we compare MultiCIDIM-DS ensembles (with or without CFC) and VFDT, the differences are obvious. Considering the number of leaves or the accuracy, we can see that the results achieved by MultiCIDIM-DS are usually significantly better than those achieved by VFDT. Even more if we consider MultiCIDIM-DS-CFC for the accuracy aspect.

Comparing MultiCIDIM-DS with MultiCIDIM-DS-CFC reveals that the accuracy is always better when using the correction filter for classification (CFC). In addition, the differences are usually significant. With these experiments we can see how using some kind of filter can improve the accuracy of the induced model.

4 Conclusion

This paper introduces a method for improving the classification (or prediction) accuracy reached by multiple classifier systems. It is based on the use of correction filters for classification (CFC). These filters are classifiers that try to learn which subspaces have been correctly learnt by the base classifiers in the ensemble. Using that information, the ensemble can improve the prediction of unseen examples.

Out aim of improving CFC involves some issues. We are working to test this approach with other kind of ensembles (online bagging and boosting, etc.), base classifiers (decision trees, SVM, etc.) and generation approaches (parallel generation). Thus we could study how much improvement can be expected from the new approach in different conditions. Another important aspect will be the study of the performance of this method in the presence of concept drift using different kinds of incremental learning algorithms to induce the CFCs.

References

1. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
2. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proc. of the 13th Int. Conf. on Machine Learning, pp. 146–148 (1996)
3. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 226–235. ACM Press, New York (2003)
4. Breiman, L.: Pasting small votes for classification in large databases and on-line. Machine Learning 36, 85–103 (1999)
5. Street, W.N., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: Proc. of the Seventh ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 377–382. ACM Press, New York (2001)
6. Oza, N.C.: Online bagging and boosting. In: Proc. of the IEEE Int. Conf. on Systems, Man and Cybernetics, pp. 2340–2345. IEEE Press, Los Alamitos (2005)
7. Hu, Q., Yu, D., Wang, M.: Constructing rough decision forests. In: Ślęzak, D., Yao, J., Peters, J.F., Ziarko, W., Hu, X. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3642, pp. 147–156. Springer, Heidelberg (2005)
8. Hu, X.: Ensembles of classifiers based on rough sets theory and set-oriented database operations. In: Proc. of the IEEE In. Conf. on Granular Computing, pp. 67–73. IEEE Press, Los Alamitos (2006)
9. Polikar, R., Udpa, L., Udpa, S., Honavar, V.: Learn++: An incremental learning algorithm for supervised neural networks. IEEE Transactions on Systems, Man, and Cybernetics 31, 497–508 (2001)

10. Fern, A., Givan, R.: Online ensemble learning: An empirical study. Machine Learning 53, 71–109 (2003)
11. Fan, W., Stolfo, S.J., Zhang, J.: The application of adaboost for distributed, scalable and on-line learning. In: Proc. of the fifth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 362–366. ACM Press, New York (1999)
12. Gama, J., Fernandes, R., Rocha, R.: Decision trees for mining data streams. Intelligent Data Analysis 10(1), 23–45 (2006)
13. Ramos-Jiménez, G., Campo-Ávila, J., Morales-Bueno, R.: Induction of decision trees using an internal control of induction. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 795–803. Springer, Heidelberg (2005)
14. Ramos-Jiménez, G., del Campo-Ávila, J., Morales-Bueno, R.: FE-CIDIM: Fast ensemble of CIDIM classifiers. International Journal of Systems Science 37, 939–947 (2006)
15. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
16. Ortega, J.: Exploiting multiple existing models and learning algorithms. In: Proc. of the Integrating Multiple Learned Models for Improving and Scaling Machine Learning Algorithms Workshop (1996)
17. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proc. of the 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 71–80. ACM Press, New York (2000)
18. Ramos-Jiménez, G., del Campo-Ávila, J., Morales-Bueno, R.: Incremental algorithm driven by error margins. In: Todorovski, L., Lavrač, N., Jantke, K.P. (eds.) DS 2006. LNCS (LNAI), vol. 4265, pp. 358–362. Springer, Heidelberg (2006)
19. del Campo-Ávila, J., Ramos-Jiménez, G., Gama, J., Morales-Bueno, R.: Improving the performance of an incremental algorithm driven by error margins. In: Intelligent Data Analysis (2007) (to appear)
20. Blake, C., Merz, C.J.: UCI repository of machine learning databases. University of California, Department of Information and Computer Science (2000)
21. Demšar, J.: Statistical comparisons of classifiers over multiple datasets. Journal of Machine Learning Research 7, 1–30 (2006)
22. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco (2000)

Combining Bagging and Random Subspaces to Create Better Ensembles

Panče Panov and Sašo Džeroski

Department of Knowledge Technologies
Jožef Stefan Institute
Ljubljana, Slovenia
{pance.panov,saso.dzeroski}@ijs.si

Abstract. Random forests are one of the best performing methods for constructing ensembles. They derive their strength from two aspects: using random subsamples of the training data (as in bagging) and randomizing the algorithm for learning base-level classifiers (decision trees). The base-level algorithm randomly selects a subset of the features at each step of tree construction and chooses the best among these. We propose to use a combination of concepts used in bagging and random subspaces to achieve a similar effect. The latter randomly select a subset of the features at the start and use a deterministic version of the base-level algorithm (and is thus somewhat similar to the randomized version of the algorithm). The results of our experiments show that the proposed approach has a comparable performance to that of random forests, with the added advantage of being applicable to any base-level algorithm without the need to randomize the latter.

1 Introduction

Random forests [1] are one of the best performing methods for constructing ensembles of classifiers. They their strength from two aspects: using random subsamples of the training data, on one hand, and randomizing the algorithm for learning base-level classifiers (decision trees). The base-level algorithm randomly selects a subset of the features at each step of tree construction and chooses the best among these.

We propose to use a combination of bagging [2] and random subspaces [3] to achieve a similar effect. In bagging, we sample the training set and generate random independent bootstrap replicates [4] (random subsamples). The replicates are then used for learning the base-level classifiers of the ensemble. In the random subspace method [3], base-level classifiers are learned from random subspaces of the data feature space. It randomly selects a subset of the features at the start and uses a deterministic version of the base-level algorithm. This procedure is somewhat similar to the randomized version of decision tree classifier used with random forests.

The advantage of this approach over random forests is that it is applicable to any base-level algorithm without the need to randomize the latter. We show that

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 118–129, 2007.

in case of decision trees as base-level algorithm our approach archives comparable results to random forests.

The rest of the paper is organized as follows. Section 2 gives an overview of randomization methods for constructing ensembles of classifiers. In Section 3 we present our combined ensemble method. Section 4 describes the experimental setup. Section 5 covers the results and the discussion and in Section 6 we present the conclusions.

2 Overview of Randomization Methods for Constructing Ensembles of Classifiers

Let us consider the standard supervised learning problem. A learning algorithm is given training examples S of the form $S = \{(X_1, Y_1), ..., (X_n, Y_n)\}$, where n is the number of training samples, for some unknown function $Y = f(X)$. The X_j values are typically vectors of the form $X_j = (x_{j1}, x_{j2}, ..., x_{jp})$ $(j = 1, 2, ..., n)$ where x_{jk} is the value of the k-th feature of example X_j and p is the number of features.

The Y values are typically drawn from a discrete set of classes $\{c_1, ..., c_K\}$ in the case of classification or from the set of numbers in the case of regression. In this work we consider only the task of classification.

The output of the learning algorithm is a classifier which is a hypothesis about the true function f. Given new examples X it predicts the corresponding Y values. An ensemble of classifiers is a set of classifiers whose individual decisions are combined by using some voting scheme (typically by weighted or unweighted voting) to classify new examples. We will denote the ensemble of classifiers by $E = \{C_1, C_2, ..., C_B\}$, where B is the number of classifiers.

Since there is no point in combining classifiers that always make similar decisions, the aim is to be able to find a set of base-level classifiers that will differ in their decisions so that they can complement each other. There are different possibilities how this can be achieved.

One possibility is to have different training sets to train the different base-level classifiers. This can be done randomly by creating random training sets from the given sample as is the case of bagging [2] or by using a random subset of features from the feature set as is the case in the random subspace method [3]. The classifiers can also be trained in series so that instances on which the preceding base-level classifiers are not accurate are given more emphasis in training the next base-level classifiers, like in boosting [5].

Another possibility of inducing ensembles of classifiers is to use randomized versions of the base-level algorithms or to use different algorithms all together to train the base-level classifiers.

Also, combined approaches exist that introduce variations in the training set (e.g bagging) and also use randomized versions of base-level learners (e.g decision trees). This is the case with random forests introduced by Breiman [1].

In this work we focus on combining bagging and the random subspace method to achieve comparable performance to random forests. In that context, in the

remaining part of this section we present an overview of bagging, the random subspace method and random forests.

2.1 Bagging

Bagging is based on the concepts of bootstrapping [4] and aggregating, and was introduced by Breiman [2]. Bootstrapping is based on random sampling with replacement. Therefore, taking a bootstrap replicate $S^i = \left(X_1^i, X_2^i, ..., X_n^i\right)$ of the training set $S = (X_1, X_2, ..., X_n)$, one can sometimes have less misleading training instances in the bootstrap training set. Consequently, a classifier constructed on such a training set may have better performance. Aggregating actually means combining of classifiers.

Often, an ensemble of classifiers gives better results than its individual base classifiers because it combines the advantages of the base-level classifiers. Bagging gives good results when unstable learning algorithms (e.g. decision trees) are used as base-level classifiers, where small changes in the training set result in largely different classifiers. The bagging algorithm is presented in Table 1.

Table 1. Bagging

Input: Training examples S, Bag size B
Output: Ensemble E
  $E \Leftarrow 0$
  for $i = 1$ to B **do**
    $S^i \Leftarrow BootstrapSample(S)$
    $C^i \Leftarrow ConstructClassifier(S^i)$
    $E \Leftarrow E \cup \left\{C^i\right\}$
  end for
  return E

When bootstrapping the training set $S = (X_1, X_2, ..., X_n)$, the probability that the training instance X_j is selected m times in a bootstrap sample S^i is given in Equation 1.

$$b\left(m|n, \frac{1}{n}\right) = C_n^m \left(\frac{1}{n}\right)^m \left(1 - \frac{1}{n}\right)^{n-m}, C_n^m = \frac{n!}{m!(n-m)!} \tag{1}$$

For large n, the binomial distribution can be approximated by the Poisson distribution, so that each instance has a probability of approximately $\frac{1}{e}$ of being left out of a bootstrap sample. Therefore, on average, approximately 37% of the instances are not present in the bootstrap sample. This means that possible outliers in the training set sometimes do not show up in the bootstrap sample. By that, better classifiers (with smaller error) may be obtained from the bootstrap sample than from the original training set.

2.2 The Random Subspace Method

The random subspace method (RSM) is an ensemble construction technique proposed by Ho [3]. In the RSM, the training set is also modified as in bagging. However, this modification is performed in the feature space (rather than example space).

Let each training example X_j in the training sample set S be a p-dimensional vector $X_j = (x_{j1}, x_{j2}, ..., x_{jp})$. In the RSM, we randomly select p^* features from the training set S, where $p^* < p$. By this, we obtain the p^* dimensional random subspace of the original p-dimensional feature space. Therefore, the modified training set $\tilde{S} = \left(\tilde{X}_1, \tilde{X}_2, ..., \tilde{X}_n\right)$ consists of p^*-dimensional training examples $\tilde{X}_j = (x_{j1}, x_{j2}, ..., x_{jp^*})\,(j = 1, 2, ..., n)$. Afterwards, base-level classifiers are constructed from the random subspaces $\tilde{S}^i$ (of the same size), $i = 1, 2, ..., B$, and they are combined by a voting scheme to obtain a final prediction. The RSM algorithm is presented in Table 2.

Table 2. Random subspace method

Input: Training examples S, Number of subspaces B, Dimension of subspaces p^*
Output: Ensemble E
  $E \Leftarrow 0$
  for $i = 1$ to B **do**
    $\tilde{S}^i \Leftarrow SelectRandomSubspace(S, p^*)$
    $C^i \Leftarrow ConstructClassifier(\tilde{S}^i)$
    $E \Leftarrow E \cup \{C^i\}$
  end for
  return E

The RSM may benefit from using both random subspaces for constructing the classifiers and aggregating the classifiers. When the number of training examples is relatively small as compared with the data dimensionality, by constructing classifiers in random subspaces one may solve the small sample problem. In this case the subspace dimensionality is smaller than in the original feature space, while the number of training objects remains the same. When the dataset has many redundant features, one may obtain better classifiers in random subspaces than in the original feature space. The combined decision of such classifiers may be superior to a single classifier constructed on the original training set in the complete feature space.

The RSM was originally developed for decision trees, but the methodology can be used to improve the performance of other unstable classifiers (e.g. rules, neural networks etc.). The RSM is expected to perform well when there is a certain redundancy in the data feature space [3]. It is noticed that the performance of the RSM is affected by problem complexity (feature efficiency, length of class boundary etc.)[6]. When applied to decision trees, the RSM is superior to a single classifier and may outperform both bagging and boosting [3].

2.3 Random Forests

Random forests, introduced by Breiman [1], have been shown to be a powerful classification and regression technique. In bagging, single models are induced over bootstrap samples of the training data, and the classification is made by using some voting scheme. Random forests is a particular implementation of bagging in which each model is a random tree. A random tree is grown according to the CART [7] algorithm with one exception: for each split, rather than considering all possible splits, only a small subset of randomly selected splits is considered (e.g. a random subset of input features), and the best split is chosen from this subset. There are two random steps when inducing the trees: the bootstrap sample for each tree, and random selection of features to split on at every node of the tree.

Let n be the number of training examples, and p the number of variables in the classifier. Let f be the number of input variables to be used to determine the decision at each node of the tree and $f << p$ (usually $f = \sqrt{p}$ or $f = \lfloor log_2(p) + 1 \rfloor$). The algorithm for constructing random forests is presented in Table 3.

Table 3. Random forest

Input: Training examples S,Bag size B, Proportion of features considered f
Output: Ensemble$E$
  $E \Leftarrow 0$
  for $i = 1$ to B **do**
    $S^i \Leftarrow BootstrapSample(S)$
    $C^i \Leftarrow BuildRandomTreeClassifier(S^i, f)$
    $E \Leftarrow E \cup \{C^i\}$
  end for
  return E

3 Combining Bagging and Random Subspace Method

In this section, we present an ensemble construction scheme in which new learning sets are generated on the basis of both bagging and random subspaces.

In this combined method, the training set is modified in two ways. First, the modification is performed in the training set by taking bootstrap replicates $S^i = \left(X_1^i, X_2^i, ..., X_n^i\right)$ of the training set $S = (X_1, X_2, ..., X_n)$. After that, a modification is performed in the feature space (like in the RSM) on every bootstrap replicate taken from the training set.

Let each example X_j^i $(j = 1, 2, ..., n; i = 1, 2, ...B)$ of a bootstrap replicate $S^i = \left(X_1^i, X_2^i, ..., X_n^i\right)$ be a p-dimensional vector $X_j^i = \left(x_{j1}^i, x_{j2}^i, ..., x_{jp}^i\right)$. We randomly select $p^* < p$ features from every bootstrap replicate X^i. By this, we obtain the p^* dimensional random subspace of the original p-dimensional feature space. Therefore, the modified training set $\tilde{S}^i = \left(\tilde{X}_1^i, \tilde{X}_2^i, ..., \tilde{X}_n^i\right)$ consists of p^*-dimensional training examples $\tilde{X}_j^i = \left(x_{j1}^i, x_{j2}^i, ..., x_{jp^*}^i\right)$ $(j = 1, 2, ..., n)$, where the p^* components x_{jk}^i $(k = 1, 2, ..., p^*)$ are randomly selected from p components

Table 4. SubBag

Input: Training examples S, Bag size B, Dimension of the subspaces p^*
Output: Ensemble E
  $E \Leftarrow 0$
  for $i = 1$ to B **do**
    $S^i \Leftarrow BootstrapSample(S)$
    $\tilde{S}^i \Leftarrow SelectRandomSubspace(S^i, p^*)$
    $C^i \Leftarrow ConstructClassifier(\tilde{S}^i)$
    $E \Leftarrow E \cup \{C^i\}$
  end for
  return E

x^i_{jk} $(j = 1, 2, ..., p)$ of the training vector X^i_j (the selection is the same for each training vector). One then constructs base-level classifiers in the random subspaces $\tilde{S}^i$ (of the same size), $i = 1, 2, ..., B$, and combines them with a voting scheme in the final prediction rule. We name this algorithm SubBag and it is presented in Table 4.

4 Implementation and Experimental Setup

For the experimental evaluation of the proposed method, we used the WEKA [8] data mining environment. In the original implementation some of the ensemble methods like bagging and random forests were already implemented. We implemented the random subspace method and our proposed combined method labeled as SubBag. The reason why the WEKA environment was chosen is that it offers a variety of learning algorithms and a well defined framework for development of new algorithms. The WEKA Experimenter was used for testing and comparing the performances of different learning algorithms.

Experiments were performed on 19 datasets taken from the UCI repository [9]. We selected datasets from the repository that have different number of training instances, different number of features, and different number of classes. A summary of the datasets used is presented in Table 5.

In this work, we compared four ensemble learning algorithms: SubBag, the random subspaces method (RSM), bagging and random forests. Experiments were conducted using three base-level algorithms: J48 which is WEKA's implementation of C4.5 [10] decision tree, JRip which is WEKA's implementation of the RIPPER [11] rule learner and the nearest neighbor algorithm IBk [12].

The comparisons of the ensemble algorithms were made with same number of classifiers (B=50). For the random subspace method and SubBag 75% of the input features were randomly selected for every classifier. With random forests, we choose $\lfloor log_2(p) + 1 \rfloor$ (where p is the number of input variables to the classifier) variables to be randomly selected for determining the spliting attribute at every node of the tree. Base-line algorithms were used with parameters as specified here: J48 algorithm was used to induce unpruned trees, JRip was used to produce unpruned rules and IBk algorithm was used with 1 nearest neighbour and the search

Table 5. Datasets used in the experiments

Dataset	Training set size	Number of features	Number of classes
arrhythmia	452	279	16
audiology	226	69	24
autos	205	25	7
cylinder-bands	540	39	2
dermatology	336	34	6
hepatitis	155	19	2
hypothyroid	3772	29	4
ionosphere	351	34	2
optdigits	5620	64	10
sick	3772	29	2
sonar	208	60	2
spambase	4601	57	2
kr-vs-kp	3196	36	2
lung-cancer	32	57	3
mfeat	2000	76	10
molecular-biology-promoters	106	58	2
mushroom	8124	22	2
splice	3190	61	3
sponge	76	45	2

was performed with linear nearest neighbour search algorithm with Eucilidian distance function. For all datasets a stratified 10-fold cross-validation was performed. A paired T-test was employed for testing the difference in performances of the algorithms on every dataset seperately. For testing whether the difference in predictive perfomance between the different methods is statisticaly significant over all datasets, we use the Wilcoxon test [13] that is sugested in [14].

5 Results and Discussion

In this section we present the results of the experiments. We first present the results of using the J48 decision tree algorithm as base-level classifier then the the JRip rule learner as a base-level classifier and nearest neighbor algorithm IBk.

In Table 6, we present the percentage of correctly classified instances using J48 as base-level classifier. A statistical comparison was performed with the SubBag method. From the results we can see that the SubBag method is on average comparable with random forests and performs better than random subspace method and bagging. Random forests preform statistically better than SubBag for the cylinder-bands and sonar datasets. Bagging performs statistically better than SubBag on one dataset, kr-vs-kp. The last column shows the percentage of correctly classified instances for the baseline algorithm J48[1]. It is obvious from the results that all ensemble methods improve the accuracy of the baseline classifier.

[1] The baseline algorithm was run with confidence parameter C=0.25 and minimum number of instances in a leaf parameter M=2.

Table 6. The accuracy of the methods using J48 as base-level classifier (the method that has the best performance for a given dataset is marked in bold, results from the performed T-test are marked with ○ if there is statistically significant improvement or with • if there is statistically significant degradation with as compared to to SubBag method)

Dataset	SubBag	Random subspaces	Bagging	Random forest	J48
arrhythmia	**72.36**	68.81	71.92	67.48 •	64.38 •
audiology	86.23	**86.70**	84.45	80.04 •	77.87 •
autos	**87.76**	86.76	86.31	86.24	81.88
cylinder-bands	64.26	64.81	62.59	**75.37** ○	57.78 •
dermatology	**97.55**	94.55 •	95.65	96.46	94.00
hepatitis	84.46	81.96	83.21	**84.54**	83.79
hypothyroid	**99.63**	99.58	99.60	99.36	99.58
ionosphere	93.46	92.32	93.17	**94.03**	91.46
optdigits	97.70	97.38	96.23 •	**98.11**	90.69 •
sick	98.67	98.99	**99.05**	98.41	98.81
sonar	80.79	76.93	77.43 •	**87.55** ○	71.17
spambase	95.37	95.28	94.76	**95.70**	92.98 •
kr-vs-kp	99.44	99.34	**99.72** ○	99.25	99.44
lung-cancer	70.83	**77.50**	70.83	67.50	**77.50**
mfeat	**84.45**	82.15 •	82.15 •	83.55	75.25 •
molecular promoters	89.55	83.73	85.45	**90.45**	80.82
mushroom	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
splice	**95.27**	94.83	94.61	93.48	94.08
sponge	93.57	93.57	93.57	**93.75**	92.50

Table 7. Comparison of the predictive performance of ensemble methods using J48 as base-level classifier with Wilcoxon test

SubBag > Random subspaces	$p = 0.040043$
SubBag > Bagging	$p = 0.001324$
SubBag > Random forests	$p = 0.489783$
SubBag > J48	$p = 0.001713$

In Table 7, we present the results of applying the Wilcoxon statistical test to the accuracies achieved using J48 as a base level classifier. We compared our proposed method with random subspaces, bagging, random forests, and the baseline method. In the results, $M1 > M2$ means that method $M1$ has better predictive performance than method $M2$. The significance is reported by adding the corresponding p-value. From the results we can see that SubBag archives statistically significant improvement over random subspaces, bagging and the baseline method. SubBag performs equally well on these datasets when compared to random forests.

In Table 8, we present the percentage of correctly classified instances using JRip as base-level classifier. In this case SubBag was statistically compared with

Table 8. The accuracy of the methods using JRip as base-level classifier (for notation see Table 6)

Dataset	SubBag	Random subspaces	Bagging	JRip
arrhythmia	**72.82**	72.14	74.81	70.80
audiology	78.28	76.05	**78.70**	72.98
autos	**83.86**	**83.86**	85.83	73.10 •
cylinder-bands	78.15	79.07	**79.26**	65.19 •
dermatology	**95.63**	94.80	92.89 •	86.88 •
hepatitis	**83.88**	83.21	80.63	78.00
hypothyroid	95.57	95.71	**99.42** ◦	99.34 ◦
ionosphere	**94.03**	93.46	92.89	89.75 •
optdigits	**98.17**	97.94	97.88	90.78 •
sick	95.97	96.24	**98.67** ◦	98.22 ◦
sonar	**86.55**	83.10	83.67	73.07 •
spambase	93.28	92.78	**94.61** ◦	92.39
kr-vs-kp	93.74	93.09	**99.47** ◦	99.19 ◦
lung-cancer	**80.83**	**80.83**	**80.83**	78.33
mfeat	83.60	**83.75**	83.60	73.15 •
molecular promoters	**88.55**	80.36	87.55	82.91
mushroom	**100.00**	**100.00**	**100.00**	**100.00**
splice	93.73	92.79	**96.11** ◦	93.70
sponge	**92.50**	**92.50**	**92.50**	**92.50**

random subspace method and bagging as well as the baseline method JRip[2]. From the results we can see that the SubBag method is comparable to the random subspace method. Bagging performs statistically better on several datasets (hypothyroid, sick, spambase, kr-vs-kp and splice) that have a large number of training examples. The baseline method also performs statistically better on three datasets (hypothyroid, sick and kr-vs-kp) possibly because of the included phase of optimization and revision of the produced rules that is a part of the RIPPER algorithm.

In Table 9 we present the results of applying the Wilcoxon statistical test to the performance of ensemble methods using JRip as a base level classifier. From the results we can see that SubBag achives statistically significant improvement over random subspaces and the baseline method. SubBag on these datasets performs statistically worse than bagging.

In Table 10, we present the percentage of correctly classified instances using IBk as base-level classifier. In this case SubBag was statistically compared with the random subspace method, bagging and the baseline method IBk [3]. From

[2] The baseline algorithm was run with the following parameters: number of folds for pruning F=3, minimum total weight of the instance in a rule N=2 and number of optimizations O=2.

[3] IBk algorithm was used with 1 nearest neighbor and the search was performed with linear nearest neighbor search algorithm with Eucilidian distance function

Table 9. Comparison of ensemble methods using JRip as base-level classifier using Wilcoxon test

SubBag > Random subspaces	$p = 0.008590$
SubBag < Bagging	$p = 0.061953$
SubBag > JRip	$p = 3.38454e^{-4}$

Table 10. The accuracy of the methods using IBk as base-level classifier (for notation see Table 6)

Dataset	SubBag	Random subspaces	Bagging	IBk
arrhythmia	**58.84**	58.40	53.55 •	52.88 •
audiology	**80.04**	78.26	76.94	77.81
autos	78.88	**79.86**	75.93	75.93
cylinder-bands	73.70	**74.44**	**74.44**	**74.44**
dermatology	**96.72**	95.65	94.54	94.54
hepatitis	**85.13**	83.88	80.63	80.63
hypothyroid	94.35	**94.41**	91.73 •	91.52 •
ionosphere	91.17	**92.60**	86.33 •	86.33 •
optdigits	98.88	**99.02**	98.61	98.61
sick	96.29	**96.58**	96.24	96.18
sonar	88.50	**89.48**	86.10	86.57
spambase	**94.07**	93.91	91.26 •	90.78 •
kr-vs-kp	**96.81**	96.62	96.62	96.28
lung-cancer	**70.83**	67.50	64.17	67.50
mfeat	**82.25**	82.20	80.10 •	80.20 •
molecular promoters	**87.82**	85.73	86.09	82.27
mushroom	**100.00**	**100.00**	**100.00**	**100.00**
splice	**92.04**	91.82	77.27 •	74.67 •
sponge	**95.00**	**95.00**	93.57	92.14

Table 11. Comparison of ensemble methods using kNN as base-level classifier using Wilcoxon test

SubBag > Random subspaces	$p = 0.331723$
SubBag > Bagging	$p = 2.7271e^{-4}$
SubBag > kNN	$p = 2.7271e^{-4}$

the results we can see that the SubBag method is comparable to the random subspace method.

In Table 11 we present the results of applying Wilcoxon statistical test to the performance of ensemble methods using IBk as a base level classifier. From the results we can see that SubBag achieves statistically significant improvement over bagging and the baseline method. SubBag on these datasets performs comparably well to random subspaces.

6 Conclusion and Further Work

In this work we present a new method for constructing ensembles of classifiers in which new learning sets are generated on the basis of bagging and random subspaces. This method was compared with other ensemble methods that use randomization to induce classifiers (bagging, random subspace method and random forests). Comparison was performed on 19 datasets from the UCI domain. As base-level classifiers we used J48 decision tree, JRip rule learner and the IBk nearest neighbor learner. By application of a paired T-test we investigated the statistical differences between these methods in terms of classification accuracy on every dataset separately. We also employed the Wilcoxon test to test the statistical significance of the methods over all datasets. The experimental results obtained show that in the case of J48 as a base-level classifier, SubBag performs comparably to random forests. The added advantage of this combination scheme is that it is applicable to any base-level algorithm without the need to randomize the algorithm itself. In case of JRip as a base-level classifier our method is statistically comparable to bagging and better than random subspace method. For the case of IBk as a base level classifier it performs better than random subspaces and bagging.

As further work, we plan to investigate the diversity of the members of the ensemble of classifiers induced by our combined approach and compare it to other ensemble methods in this respect. Another possibility to investigate is using a different combination of bagging and random subspaces method (e.g. bags of RSM ensembles and RSM ensembles of bags). Finally, a comparison of bagged ensembles of randomized base-level classifiers (e.g. a randomized version of JRip) would be of interest.

Acknowledgments. This work was supported by the EU FP6-516169 project "Inductive Queries for Mining Patterns and Models". The authors would like to thank Dragi Kocev and Ivica Slavkov for providing valuable suggestions for improving this manuscript.

References

1. Breiman, L.: Random forests. Mach. Learn. 45(1), 5–32 (2001)
2. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
3. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
4. Efron, B., Tibshirani, R.J.: An introduction to the Bootstrap. Monographs on Statistics and Applied Probability, vol. 57. Chapman and Hall (1993)
5. Schapire, R.E.: The strength of weak learnability. Machine Learning 5, 197–227 (1990)
6. Ho, T.K.: Complexity of classification problems and comparative advantages of combined classifiers. In: MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems, London, UK, pp. 97–106. Springer, Heidelberg (2000)

7. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA (1984)
8. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques (Morgan Kaufmann Series in Data Management Systems), 2nd edn. Morgan Kaufmann, San Francisco (2005)
9. Newman, D.J., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
10. Quinlan, J.R.: C4.5: programs for machine learning. Kaufmann Publishers Inc., San Francisco, CA, USA (1993)
11. Cohen, W.W.: Fast effective rule induction. In: Prieditis, A., Russell, S. (eds.) Proc. of the 12th International Conference on Machine Learning, Tahoe City, CA, pp. 115–123. Morgan Kaufmann, San Francisco (1995)
12. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Mach. Learn. 6(1), 37–66 (1991)
13. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics 1, 80–83 (1945)
14. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30 (2006)

Two Bagging Algorithms with Coupled Learners to Encourage Diversity*

Carlos Valle[1], Ricardo Ñanculef[1], Héctor Allende[1], and Claudio Moraga[2,3]

[1] Universidad Técnica Federico Santa María,
Departamento de Informática, CP 110-V Valparaíso, Chile
{cvalle,jnancu,hallende}@inf.utfsm.cl
[2] European Centre for Soft Computing 33600 Mieres, Asturias, Spain
[3] Dortmund University, 44221 Dortmund, Germany
mail@claudio-moraga.eu

Abstract. In this paper, we present two ensemble learning algorithms which make use of boostrapping and out-of-bag estimation in an attempt to inherit the robustness of bagging to overfitting. As against bagging, with these algorithms learners have visibility on the other learners and cooperate to get diversity, a characteristic that has proved to be an issue of major concern to ensemble models. Experiments are provided using two regression problems obtained from UCI.

Keywords: Ensemble Methods, Bagging, Ensemble Diversity, Regression Estimators, Neural Networks, Out-of-bag Estimation.

1 Introduction

Ensemble algorithms are general methods to improve the performance of a base learning algorithm, which are actually applied to a great variety of problems including classification, clustering and regression, which is the focus of this paper. Given a set of examples $S = \{(x_k, y_k); k = 1, \ldots, m\}$, an ensemble algorithm generates a set of predictors $H = \{f_1, f_2, \ldots, f_n\}$ by training a base learner L. Then it builds a new predictor F by aggregating the predictors in H, using for example linear combinations.

A well-studied method of this family is *Bagging*, introduced by Breiman in [2]. In Bagging, the set of predictors is generated by training L with a sequence of bootstrap datasets $S_1, S_2, \ldots, S_n$, randomly drawn from the original training set S. These predictors are then combined (averaged in regression) to obtain what is called *the bagged predictor*. A commonly accepted explanation for the ability of Bagging to improve the generalization performance of the base model, is that it could effectively reduce its variance. Recent works [11] [12] have however shown that Bagging can work without reducing variance. They argue that the

* This work was supported in part by Research Grant Fondecyt (Chile) 1040365, 7060040 and 1070220. Partial support was also received from Research Grant 2407265 DGIP-UTFSM (Chile).

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 130–139, 2007.

action of Bagging is to control the effective influence of training examples in the estimation process, through an adequate sampling of them. This argument is consistent with some attempts to relate the stability of the base algorithm L and the stability of the bagged algorithm [13] [8].

It is interesting to note that the benefits of the random sampling, characteristic of bagging, could be applied to other ensemble algorithms. For example in [10], Friedman proposes to incorporate uniform random subsampling, at each stage of the *Gradient Boosting* algorithm. Another interesting observation with bagging is that roughly a fraction 0.368 of the training examples do not appear in a particular bootstrap sample. As noted by Breiman [3], these *out-of-bag* examples are unused test examples for the predictor constructed with the corresponding sample, and can be used to make accurate estimates of quantities of interest. In [4] for example, Breiman proposes a new form of bagging called *Iterated Bagging* that exploits this observation.

In this paper we present two algorithms which make use of random sampling and out-of-bag estimates. We start by presenting in section 2, an ensemble learning algorithm where a set of learners is trained to cooperate and to obtain diverse generalization patterns, that is a characteristic that has proved to be an issue of major concern to ensemble models [5]. In section 3 we show how to incorporate random sampling to this algorithm in an attempt to inherit the robustness of bagging to overfitting. As against bagging, with this algorithm learners have visibility on predictions of the other learners and cooperate to get diversity. In section 4 we introduce the use of out-of-bag estimates on the algorithm from the previous section. Resultant algorithm is similar to Iterated Bagging, but uses different targets at each stage and iterations are carried out on the same layer of learners. In the final section we provide experimental results on two regression datasets obtained from UCI[1].

2 Diversity and Negative Correlation Learning

Several works have shown that a key characteristic of ensemble models, to get real improvements of combining predictors, is diversity of their component members. For regression estimation, a way to quantify diversity is the so called *Ambiguity Decomposition*. Let F be, an ensemble obtained as a convex combination of n predictors, that is

$$F(x) = \sum_{i=1}^{n} w_i f_i(x) \tag{1}$$

Hence[1],

$$(F - y)^2 = \sum_{i=1}^{n} w_i (y - f_i)^2 - \sum_{i=1}^{n} w_i (f_i - F)^2 \tag{2}$$

This decomposition states that the quadratic loss of the ensemble not only depends on the individual errors but also on the variability of the individual predictions f_i around the ensemble prediction F. Ambiguity decomposition suggests

[1] To simplify notation we usually replace $F(x)$ and $f_i(x)$ by F and f_i respectively.

that individual learners should be trained considering information about the other learners to get *diversity*, which can be quantified and measured as the second term in (2). For example, in the *Negative Correlation Algorithm* (NC), described in [17], the i-th learner is trained considering the objective function

$$\text{NCError}_i = (y - f_i)^2 - \lambda (f_i - F)^2 \tag{3}$$

where the parameter λ weights the importance of the diversity component. As noted in [5], if the predictors in the ensemble are uniformly weighted, we obtain that

$$\text{NCError}_i = (y - f_i)^2 + \lambda \sum_{j \neq i} (f_i - F)(f_j - F) \tag{4}$$

If $\lambda = 0$ each learner is trained independently. In [5] a theoretical argument is presented to choose λ according to $\lambda = 2 \cdot \gamma \cdot (1 - 1/n)$ where n is the size of the ensemble and the value of $\gamma \in [0, 1]$ is problem dependent.

The authors have previously shown in [14] that and alternative decomposition of the quadratic loss of F is the following

$$(F - y)^2 = \sum_{i=1}^{n} w_i^2 (y - f_i)^2 + \sum_{i=1}^{n} \sum_{j \neq i} w_i w_j (f_i - y)(f_j - y) \tag{5}$$

It should be noted that in this decomposition, diversity (second term of the right hand side) is measured in terms of error correlations instead of hypothesis deviations around the ensemble. If the ensemble is uniformly weighted, it is hence reasonable to train each learner with the training function

$$\text{NegCorrErr}_i = (y - f_i)^2 + \lambda \sum_{j \neq i} (f_i - y)(f_j - y) \tag{6}$$

where $\lambda > 0$ controls the tradeoff between diversity and individual accuracy.

1: Let $S = \{(x_k, y_k); k = 1, \ldots, m\}$ be a training set.
2: Let $\{f_i; i = 1, \ldots, n\}$ be a set of n learners and f_i^t the function implemented by the learner f_i at time $t = 0, \ldots, T$.
3: Make one epoch on the learner f_i with the training set S and the squared loss function to obtain the initial functions f_i^0.
4: **for** $t = 1$ to T
5: **for** $i = 1$ to M
6: Make one epoch on the learner f_i with S and the loss function

$$\text{NegCorrErr}_i^t(f_i(x), y) = (y - f_i(x))^2 + \lambda \sum_{j \neq i} (f_i(x) - y)\left(f_j^{t-1}(x) - y\right)$$

7: **end for**
8: Set the ensemble predictor at time t to be $F^t(x) = 1/n \sum_{i=1}^{n} f_i^t(x)$
9: **end for**

Fig. 1. Algorithm I

In Algorithm (I) the learners of the ensemble are synchronously trained based on the loss function (6). Note that at iteration t the diversity component is computed using the predictions of the other learners obtained at the previous iteration. Hence the algorithm consider one stage where the learners share information about their predictions and one stage where the learners are adjusted.

3 Bootstrapping Negative Correlation

In [2] Breiman presents bagging as a procedure capable to reduce the variance of regression predictors. In [12] and [11] however, Grandvalet provides experimental evidence supporting the hypothesis that bagging stabilizes prediction by controlling the influence of training examples, and can reduce the testing error without reducing variance. According to this explanation, the action of bagging is to equalize the effective influence of each example in the estimation process carried out by the learning algorithm, in a way such that highly influential points (the so called leverage points) are down-weighted. As stated by Grandvalet, the positive or negative consequences of this action depends on the goodness/badness of leverage. Since in most situations, leverage points are badly influential, for example they are outliers, bagging can improve generalization by making robust an unstable base learner. From this point of view, bootstrapping the base learner has an effect similar to *robust M-estimators* where the influence of sample points is (globally) bounded using robust loss functions.

The latter explanation about the success of bagged predictors is consistent with others studies that make a connection between the (algorithmic) stability of the base algorithm and the stability of the bagged algorithm [13] [8]. Roughly speaking, certain resampling strategies allow poorly stable algorithms become strongly stable. Strong algorithms provide fast rates of convergence from the empirical error to the true expected prediction error.

The key fact in the previous analyses is that bootstrapping allows some points affect only a subset of learners in the ensemble. It seems counterintuitive however, that bootstrap has the ability to selectively reduce the influence of leverage points, in particular badly influential points, since in uniform resampling all the points have the same probability of being selected. The explanation is that leverage points are usually isolated in the feature space. To remove the influence of a leverage point it is enough to eliminate this point from the sample but to remove the influence of a non-leverage point we must in general remove a group of observations. In other words, non-leverage points correspond to local patterns represented by more than one example. Now, the probability that a group of size K be completely ignored by bagging is $(1 - K/m)^m$ which decays exponentially with K. For $K = 2$ for example $(1 - K/m)^m \sim 0.14$ while $(1 - 1/m)^m \sim 0.368$. This means that bootstrapping allows the ensemble predictions (and hence the learning process itself) depend mainly on "common" examples, which in turns allows to get a better generalization.

We have seen that ensemble learning can be achieved by training each learner with an objective function that represents a compromise between the individual accuracy and diversity in terms of error correlations (algorithm I). This seems reasonable, because this explicitly encourages mutual cooperation between the learners. It should be noted however, that bagging does not provide such kind of mutual cooperation. On the other hand, algorithm (I) not only does not prevent the potentially bad effect of leverage points but propagates this effect through the whole ensemble. When a point induces a significant error for a predictor, the other predictors will try to compensate this error. In other words, the ensemble predictions could strongly depend on "unusual" examples, then compromising the generalization performance of the ensemble.

To combine the key characteristics of both algorithms: mutual cooperation of algorithm (I) and robustness of bagging due to the ability of bootstrapping to equalize the influence of the examples, we propose to modify algorithm (I) in a way such that each learner works with a bootstrap sample of the original training set. The result is exposeddescribed as algorithm (II).

1: Let $S = \{(x_k, y_k); k = 1, \ldots, m\}$ be a training set.
2: Let $\{f_i; i = 1, \ldots, n\}$ be a set of n learners and f_i^t the function implemented by the learner f_i at time $t = 0, \ldots, T$.
3: Generate n bootstrap samples S^i, $i = 1, \ldots, n$ from S.
4: Make one epoch on the learner f_i with the training set S_i and the squared loss function to obtain the initial functions f_i^0.
5: **for** $t = 1$ to T
6: **for** $i = 1$ to M
7: Make one epoch on the learner f_i with S_i and the loss function

$$\text{NegCorrErr}_i^t(f_i(x), y) = (y - f_i(x))^2 + \lambda \sum_{j \neq i} (f_i(x) - y)\left(f_j^{t-1}(x) - y\right)$$

8: **end for**
9: Set the ensemble predictor at time t to be $F^t(x) = 1/n \sum_{i=0}^{n-1} f_i^t(x)$
10: **end for**

Fig. 2. Algorithm II: Bootstrapped Negative Correlation

With this algorithm, each learner is iteratively trained on its own training set, as with bagging, but at each iteration it takes into account the diversity component used within algorithm (I). Hence, this algorithm can be viewed as a coupled bagging, where each leaner has visibility on the predictions of the other learners. However, since each learner is trained only with the examples contained in S^i, highly influential points have a restricted effective influence on the ensemble, which should allow to inherit the generalization power of bagging.

4 Using Out-of-Bag Residuals to Compute Diversity

At the t-th iteration of algorithm (I), each learner f_i of the ensemble is trained to minimize over the training set S the following objective function

$$\text{NegCorrErr}_i^t(f_i(x), y) = (y - f_i(x))^2 + \lambda \sum_{j \neq i} (f_i(x) - y) \left(f_j^{t-1}(x) - y\right) \quad (7)$$

It should be noted that for a single fixed example (x_k, y_k) optimality implies

$$2(y_k - f_i^t(x_k)) + \lambda \sum_{j \neq i} (f_i^t(x_k) - y_k)(f_j^{t-1}(x_k) - y_k) = 0 \quad (8)$$

that is $f_i^t(x_k) = \tilde{y}_k$, where

$$\tilde{y}_k = y_k + \lambda' r_i^{t-1}(k) \quad (9)$$

where in turns $\lambda' = \lambda/2$ and

$$r_i^{t-1}(k) = \sum_{j \neq i} \left(f_j^{t-1}(x_k) - y_k\right) \quad (10)$$

Note that $r_i^{t-1}(k)$ corresponds (except for a constant) to the averaged error of $F^{(i)}$ in the prediction of y_k, where $F^{(i)}$ is the set of predictors in F, with the i-th one removed. In equations, $r_i^{t-1}(k) = (n-1)\left(F^{(i)}(x_k) - y_k\right)$.

Equation (9) shows that training f_i with S and the objective function (7) is equivalent to train f_i with the standard squared loss, and the set of modified examples $\tilde{S} = \{(x_k, \tilde{y}_k); k = 1, \ldots, m\}$, where $\tilde{y}_k$ is defined in equation (9).

Note now, that $\tilde{y}_k$ is the original target y_k with the additional additive term $r_i^{t-1}(k)$ representing the residual of the ensemble without the learner f_i in predicting the training example y_k.

It is well-known however that training set residuals have a significant tendency to be too small, since the model is being trained to make these residuals as small as possible. This implies that they are in general overly optimistic estimates of the generalization error of the model. In general, for typical loss functions l, the empirical error $\hat{R} = 1/m \sum_k l(f(x_k, y_k))$ tends to be a down biased estimator of the true generalization error $R = E[l(f(x), y)]$, and hence the pseudo residual $\delta\hat{R}/\delta f$ is not a good estimator of the true prediction error landscape at f.

A more realistic estimator of the generalization error, widely used in the machine learning community, is the leave-one-out cross validation estimate. For any example (x_k, y_k) let be f^{-k} the function obtained training an algorithm with the training set without this example. The leave-one-out cross validation estimate of the generalization error of the algorithm is then defined to be

$$\hat{R}_{loo} = \frac{1}{m} \sum_{k=1}^{m} l(f^{-k}(x_k), y_k) \quad (11)$$

Hence, a more realistic estimator of the residual of $F^{(i)}$ at a point (x_k, y_k) could be obtained by computing the error of $F^{(i)}$ in this point after training with a sample without (x_k, y_k). If we repeat this procedure M times, the mean of these residuals gives a unbiased estimate of the bias of the model $F^{(i)}$ in this point. Computing requirements of this procedure are of course large.

Consider however algorithm (II). With this algorithm, each learner works with a bootstrap sample S_i of the original sample S. Approximately a fraction $(1 - 1/m)^m$ (~ 0.368) of the examples of S does not appear in the particular sample S_i, and hence these examples can be considered test examples for the learner i, which can be used to make accurate estimates of quantities of interest. This idea has been introduced by Breiman in [3] under the name of *out-of-bag* estimation. In [4] for example, Breiman proposes a new form of bagging called *Iterated Bagging* designed to simultaneously reduce regressors bias and variance. The algorithm works in stages - the first stage is bagging. A second stage of bagging is then carried out (with new learners) using the out-of-bag estimates of the residuals of the first stage as targets. This procedure is iterated until a stopping criterion is satisfied. Breiman concludes that experimental results of iterated bagging are comparable to the best results obtained using (highly tuned) *Support Vector Regression Machines* [6] while in [15] Suen et al. show that it exhibits better generalization patterns that *Stochastic Gradient Boosting*, a technique introduced by Friedman in [10] consisting basically in the randomization of *Gradient Boosting* [9].

Now, we propose to compute the residuals $r_i^{t-1}(k)$ of equation (9) using the out-of-bag procedure of Breiman, that is, instead of computing the residual $r_i^{t-1}(k)$ considering $F^{(i)}$ we look for the learners f_j in $F^{(i)}$ for which the example (x_k, y_k) is not contained in S^j - the bootstrap sample corresponding to this learner. The out-of-bag estimate of $r_i^{t-1}(k)$ is hence defined to be

$$\hat{r}_i^{t-1}(k) = \sum_{j \in C_{ik}} \left(f_j^{t-1}(x_k) - y_k \right) \tag{12}$$

where $C_{ik} = \{j \neq i / (x_k, y_k) \notin S_j\}$. With this estimate of the residual, the target used to train the learner at iteration t results

$$\begin{aligned} \hat{y}_k &= y_k + \lambda' \hat{r}_i^{t-1}(k) \\ &= y_k + \lambda' \sum_{j \in C_{ik}} \left(f_j^{t-1}(x_k) - y_k \right) \end{aligned} \tag{13}$$

It should be noted that training the learners with the set of modified examples $\hat{S} = \{(x_k, \hat{y}_k); k = 1, \ldots, m\}$ and the squared loss function, results in turns equivalent to optimize at each iteration the following objective function

$$\text{oobErr}_i^t(f_i(x), y) = (y - f_i(x))^2 + \lambda \sum_{j \in C_i(x,y)} (f_i(x) - y) \left(f_j^{t-1}(x) - y \right) \tag{14}$$

where $C_i(x, y) = \{j \neq i / (x, y) \notin S_j\}$. Note that now, the diversity component of the objective function, that encourages negative error correlation on the ensemble, is computed considering only the learners that have not seen the example

1: Let $S = \{(x_i, y_i); i = 1, \ldots, m\}$ be training set.
2: Let f_i $i = 0, \ldots, n-1$ be a set of n learners and f_i^t the function implemented by the learner f_i at time $t = 0, \ldots, T$.
3: Generate n bootstrap samples S^i, $i = 1, \ldots, n$ from S.
4: **for** $t = 1$ to T
5: Make one epoch on the learner f_i with the learning function $\text{oobErr}_i^t(f_i(x), y)$ as defined in equation (14), and the set of examples S^i.
6: Set the ensemble predictor at time t to be $F^t(x) = 1/n \sum_{i=0}^{n-1} f_i^t(x)$
7: **end for**

Fig. 3. Algorithm III: Self-iterated bagging with negative correlation

(x, y). This suggests that a possibly more realistic estimate of the amount of diversity existing in the ensemble for prediction purposes might be used. The resulting procedure is summarized as algorithm (III).

Two fundamental differences distinguish the proposed algorithm from Iterated Bagging. At each iteration, algorithm (III) uses the same set of learners - just a new training epoch on each learner is carried out considering the residuals of the previous iteration. On the other hand, learners are not designed to approximate the residuals but to optimize the objective function (14) which in turns is equivalent to learn the modified targets (13) computed using the out-of-bag estimates of $r_i^{t-1}(k)$. This in fact makes possible to use the same set of learners at each stage. Considering these point of views, algorithm (III) can be considered a self-iterated bagging with coupled learners, that is learners that cooperate to get diversity.

5 Experimental Results

In this section we present results of empirical studies to evaluate the proposed algorithms. In the whole set of experiments, two real and well-known data sets were used, namely *Boston* and *NO2*. A detailed description of these data sets can be obtained from [1] and [16] respectively. For comparison purposes, five algorithms will be evaluated: Bagging, Negative Correlation (NC), algorithm (I), algorithm (II), and algorithm (III). In addition, neural networks with two sigmoidal hidden units and trained with standard backpropagation were employed as base learners. For each experiment, t-student confidence intervals will be reported with a significance of 0.02 obtained after 10 simulations. The estimation process is carried out with a 75% of the available observations and testing with the rest 25%.

The NC algorithm depends on the parameter γ defined in section 2. Similarly algorithms (I), (II), and (III) depend on parameter λ. Due to space limitations, we cannot report exhaustively the results obtained with different constants, and hence we only present the best results obtained with the parameters which show the lower testing error (we don't use a validation set).

Table 1. Experimental results on the *Boston Housing* dataset

N	NC	Bagging	Alg.I	Alg.II	Alg.III
Training Set					
20	8.44 ∼ 8.69	10.66 ∼ 10.79	10.98 ∼ 11.23	9.22 ∼ 9.33	8.76 ∼ 8.88
30	7.99 ∼ 8.26	10.58 ∼ 10.72	11.11 ∼ 11.46	9.27 ∼ 9.36	9.12 ∼ 9.36
40	7.98 ∼ 8.19	10.53 ∼ 10.65	11.95 ∼ 11.19	8.89 ∼ 8.97	8.65 ∼ 8.75
Testing Set					
20	13.79 ∼ 14.30	13.97 ∼ 14.93	14.89 ∼ 15.38	12.50 ∼ 12.90	12.75 ∼ 13.22
30	13.19 ∼ 13.61	13.89 ∼ 14.86	15.17 ∼ 15.76	13.02 ∼ 13.46	12.69 ∼ 13.13
40	13.87 ∼ 14.22	13.73 ∼ 14.19	14.86 ∼ 15.36	12.81 ∼ 13.25	12.89 ∼ 13.36

Table 2. Experimental results on the *NO2* dataset

N	NC	Bagging	Alg.I	Alg.II	Alg.III
Training Set					
20	.2507 ∼ .2608	.2457 ∼ .2470	.3041 ∼ .3220	.2290 ∼ .2306	.2327 ∼ .2345
30	.2431 ∼ .2538	.2451 ∼ .2466	.3058 ∼ .3237	.2336 ∼ .2346	.2333 ∼ .2349
40	.2388 ∼ .2467	.2453 ∼ .2467	.3044 ∼ .3216	.2306 ∼ .2316	.2338 ∼ .2306
Testing Set					
20	.2832 ∼ .2970	.2486 ∼ .2531	.2990 ∼ .3171	.2419 ∼ .2459	.2441 ∼ .2466
30	.2973 ∼ .3137	.2469 ∼ .2511	.3011 ∼ .3189	.2451 ∼ .2485	.2501 ∼ .2541
40	.2905 ∼ .3005	.2473 ∼ .2517	.2923 ∼ .3099	.2415 ∼ .2462	.2458 ∼ .2500

Table (1) shows confidence intervals for the *mean squared error* (*mse*) of the algorithms versus the number of learners (N) in the ensemble, obtained with the *Boston Housing* dataset. For this problem, the best value of γ, for the NC algorithm was found at $\gamma = 0.5$. For algorithms (I), (II), and (III), on the other hand, the best values of λ were found at $\lambda = 0.8$, $\lambda = 0.7$ and $\lambda = 0.8$ respectively.

Table (2) shows confidence intervals for the *mse* of the algorithms versus the number of learners in the ensemble, obtained with the *NO2* dataset. For this problem, the best value of γ for the NC algorithm was found at $\gamma = 0.5$. The best value of λ was found at 0.6 for the three algorithms (I), (II), and (III).

6 Conclusions and Final Remarks

In this paper we have presented two algorithms that explicitly encourage diversity and use bootstrapped samples in order to improve generalization performance. The algorithms are constructed using ideas from the negative correlation algorithm and bagging. Experiments with neural networks ensembles in two real data sets allow us to conclude that the algorithm (II) exhibits the best generalization patterns between the five simulated algorithms, except only for one setting (30 learners with the *Boston Housing* problem) in which the algorithm (III) shows the best result. Algorithm (III) also tends to show better testing results than Bagging, Negative Correlation and algorithm (I). In the *NO2* dataset for example,

the relative improvement with respect to Negative Correlation is significant. This latter algorithm also shows the worst difference between training and testing performance, in both datasets. All of these observations confirm the hypothesis that bootstrapping can help to obtain more robust estimators to overfitting. The use of out-of-bag estimates of the residuals however, does not improve the performance of algorithm (II). This could be due to the number of learners in the ensemble, which makes the out-of-bag estimates rely on a small subset of predictors. Future work has then to include a more exhaustive experimental analysis of the behavior of the proposed algorithms with a greater number of predictors. An alternative approach could be to combine the out-of-bag estimates of the residuals with the estimates computed as in algorithm (II), in an approach similar to the 0.632 bootstrap estimator of the prediction error [7].

References

1. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)
2. Breiman, L.: Bagging predictors. Machine Learning 26(2), 123–140 (1996)
3. Breiman, L.: Out-of-bag estimation. Technical report, Statistics Department, University of California (1997)
4. Breiman, L.: Using iterated bagging to debias regressions. Machine Learning 45(3), 261–277 (2001)
5. Brown, G.: Diversity in Neural Network Ensembles. PhD thesis, School of Computer Science, University of Birmingham (2003)
6. Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A., Vapnik, V.: Support vector regression machines. In: Advances in Neural Information Processing Systems, vol. 9, p. 155. The MIT Press, Cambridge (1997)
7. Efron, B., Tibshirani, R.: An Introduction to the Bootstrap. Chapman & Hall (1993)
8. Elisseeff, A., Evgeniou, T., Pontil, M.: Stability of randomized learning algorithms. J. Machine Learning Research 6, 55–79 (2005)
9. Friedman, J.: Greedy function approximation: A gradient boosting machine. Annals of Statistics 29(5) (2001)
10. Friedman, J.: Stochastic gradient boosting. Computational Statistics and Data Analysis 38(4), 367–378 (2002)
11. Grandvalet, Y.: Bagging can stabilize without reducing variance. In: Dorffner, G., Bischof, H., Hornik, K. (eds.) ICANN 2001. LNCS, vol. 2130, pp. 49–56. Springer, Heidelberg (2001)
12. Grandvalet, Y.: Bagging equalizes influence. Machine Learning 55(3), 251–270 (2004)
13. Poggio, T., Rifkin, R., Mukherjee, S.: Bagging regularizes. Technical Report 214/AI Memo 2002-003, MIT CBCL (2002)
14. Ñanculef, R., Valle, C., Allende, H., Moraga, C.: Ensemble learning with local diversity. In: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4131, pp. 264–273. Springer, Heidelberg (2006)
15. Suen, Y., Melville, P., Mooney, R.: Combining bias and variance reduction techniques for regression. In: Proceedings of the 16th European Conference on Machine Learning, pp. 741–749 (2005)
16. Vlachos, P.: StatLib datasets archive (2005)
17. Yao, X., Lui, Y.: Ensemble learning via negative correlation. Neural Networks 12(10), 1399–1404 (1999)

Relational Algebra for Ranked Tables with Similarities: Properties and Implementation*

Radim Belohlavek[1,3], Stanislav Opichal[2], and Vilem Vychodil[3]

[1] Dept. Systems Science and Industrial Engineering
T. J. Watson School of Engineering and Applied Science
Binghamton University–SUNY, PO Box 6000, Binghamton, NY 13902–6000, USA
rbelohla@binghamton.edu
[2] PIKE ELECTRONIC, Ltd., Czech Republic
sopichal@pikeelectronic.com
[3] Dept. Computer Science, Palacky University, Olomouc
Tomkova 40, CZ-779 00 Olomouc, Czech Republic
vilem.vychodil@upol.cz

Abstract. The paper presents new developments in an extension of Codd's relational model of data. The extension consists in equipping domains of attribute values with a similarity relation and adding ranks to rows of a database table. This way, the concept of a table over domains (i.e., relation over a relation scheme) of the classical Codd's model extends to the concept of a ranked table over domains with similarities. When all similarities are ordinary identity relations and all ranks are set to 1, our extension becomes the ordinary Codd's model. The main contribution of our paper is twofold. First, we present an outline of a relational algebra for our extension. Second, we deal with implementation issues of our extension. In addition to that, we also comment on related approaches presented in the literature.

1 Introduction

1.1 Motivation and Outline of the Paper

Most of the current database systems are based on the well-known Codd's relational model of data: "A hundred years from now, I'm quite sure, database systems will still be based on Codd's relational foundation." [9, p. 1]. Main virtues of Codd's model are due to the reliance of the model on a simple yet powerful mathematical concept of a relation and first-order logic: "The relational approach really is rock solid, owing (once again) to its basis in mathematics and predicate logic." [9, p. 138].

Our paper is concerned with a particular extension of the relational model which is concerned with imprecision and uncertainty. Management of uncertainty and imprecision is one of the six currently most-important research directions

* Supported by by grant No. 1ET101370417 of GA AV ČR and by institutional support, research plan MSM 6198959214.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 140–151, 2007.

Table 1. Ranked data table over domains with similarities

$\mathcal{D}(t)$	*name*	*age*	*education*
1.0	Adams	30	Comput. Sci.
1.0	Black	30	Comput. Eng.
0.9	Chang	28	Accounting
0.8	Davis	27	Comput. Eng.
0.4	Enke	36	Electric. Eng.
0.3	Francis	39	Business

$$n_1 \approx_n n_2 = \begin{cases} 1 & \text{if } n_1 = n_2 \\ 0 & \text{if } n_1 \neq n_2 \end{cases}$$

$$a_1 \approx_a a_2 = s_a(|a_1 - a_2|) \quad \text{with scaling} \quad s_a : \mathbb{Z}^+ \to [0,1]$$

$\approx_e$	A	B	CE	CS	EE
A	1	.7			
B	.7	1			
CE			1	.9	.7
CS			.9	1	.6
EE			.7	.6	1

proposed in the report from the Lowell debate by 25 senior database researchers [1]: "... current DBMS have no facilities for either approximate data or imprecise queries." Similarity, approximate matches, similarity-based queries, and related issues are the main motivations for our extension of the relational model. These issues are not new and have been approached in numerous papers in the past. The issues result in situations when one considers similarity of elements of domains rather than exact equality, i.e. when it is desirable to consider degrees of similarity rather than just "equal" and "not equal". For example, consider attribute age. The corresponding domain consists of positive integers. One might be interested in all persons in a given database with age equal to 30. Such a query is typical in the classical relational model (in terms of relational algebra: selection of all tuples with attribute age = 30). One might, however, be also interested in all persons in the database with age approximately 30. Intuitively, a person with age 30 satisfies this query completely (degree of satisfaction is 1.0), a person with age 29 satisfies this query rather well (degree of satisfaction is, say, 0.9), a person with age 25 satisfies this query but only to a small degree (say, 0.2), etc. The above degrees are, in fact, degrees of similarity, see [23], of the actual age to the reference age 30, i.e. 1.0 is a degree of similarity of 30 to 30, 0.9 is a degree of similarity of 29 to 30, 0.2 is a degree of similarity of 25 to 30. Of course, the degrees depend on how the similarity relation is defined. The above example, however simple, clearly demonstrates that taking similarity into account leads to qualitatively new features in querying and manipulation of data. Our attempt in previous papers as well as in this paper is to develop systematically an extension of the classical Codd's relational model which would play the same role in case when similarities are considered as the ordinary Codd's model plays in the classical case.

The main concept in our approach is that of a ranked data table (relation) over domains with similarities, see Tab. 1. This concept is our counterpart to the concept of a data table (relation) over domains of a classical relational model. A ranked data table over domains with similarities consists of three parts: data

table (relation), domain similarities, and ranking. The data table (right top table in Tab.1) coincides with a data table of a classical relational model. Domain similarities and ranking are what makes our model an extension of the classical model. The domain similarities (bottom part of Tab.1) assign degrees of similarity to pairs of values of the respective domain. For instance, a degree of similarity of "Computer Science" and "Computer Engineering" is 0.9 while a degree of similarity of "Computer Science" and "Electrical Engineering" is 0.6. The ranking assigns to each row (tuple) of the data table a degree of a scale bounded by 0 and 1 (left top table in Tab.1), e.g. 0.9 assigned to the tuple $\langle \text{Chang}, 28, \text{Accounting} \rangle$. The ranking allows us to view the ranked table as an answer to a similarity-based query (rank = degree to which a tuple matches a query). For instance, the ranked table of Tab.1 can result as an answer to query "show all candidates with age about 30". In a data table representing stored data (i.e. prior to any querying), ranks of all tuples of the table are equal to 1. Therefore, the same way as tables in the classical relational model, ranked tables represent both stored data and outputs to queries. This is an important feature of our model.

We use fuzzy logic as our formal framework. We use a formal system of first-order fuzzy logic the same way as the system of first-order classical logic is used in the classical relational model. This way, our model keeps the user-friendly symbolical character of the classical model and adds a quantitative layer which takes care of the management of uncertainty. This is an important distinction from other "fuzzy approaches" to the relational model which, from our point of view, are often *ad-hoc*.

In our previous papers, we developed selected issues within the framework of our extension, e.g., functional dependencies, computation of non-redundant sets of functional dependencies, Armstrong-like axiomatization, and related issues are studied in [2,3,4,5].

In the present paper, we focus on relational algebra for our extension of the relational model. In particular, we present an overview of operations of the relational algebra, present illustrative examples and selected results on properties of the relational algebra. In addition to that, we focus on the problem of implementation of our extension. Section 1.2 briefly reviews related approaches. Section 1.3 summarizes preliminaries from fuzzy logic. In Section 2.1 we introduce our model. Section 2.2 presents relational algebra and related results. Section 2.3 deals with implementation of our relational model. Section 3 outlines future research.

1.2 Related Approaches

The first paper on a "fuzzy approach" to the relational model is [7]; [6] provides an overview with many references. We found over 100 contributions related to "fuzzy approach" to the relational model. A main feature of almost all of the approaches is that they are *ad-hoc* in that an analogy of a clear relationship between a relational model and first-order fuzzy logic is missing in the approaches

which leads to the impression of arbitrariness of the approaches. This is partly because fully fledged logical calculi have not been developed until quite recently, see e.g. [12,13]. On the other hand, several ideas including some of those presented in our paper were already discussed in the literature. For instance, the idea of considering domains with similarity relations goes back to [7]. The idea of assigning ranks to tuples appeared in [21] although with not quite a clear meaning of ranks ("fuzzy measure of association among a set of domain values" [21]).

1.3 Preliminaries

We use fuzzy logic to represent and manipulate truth degrees of propositions like "u is similar to v". Moreover, we need to process (aggregate) the degrees. For instance, consider a query "show all candidates which are about 30 years old and a degree in specialization similar to Computer Science". According to Tab. 1, Davis satisfies subqueries concerning age and education to degrees 0.8 and 0.9, respectively. Then, we combine the degrees using a fuzzy conjunction connective $\otimes$ to get a degree $0.8 \otimes 0.9$ to which Davis satisfies the conjunctive query.

When using fuzzy logic, we have to pick an appropriate scale L of truth degrees (which serve e.g. as grades for evaluating similarity of two objects) and appropriate fuzzy logic connectives (conjunction, implication, etc.). We follow a modern approach in fuzzy logic in that we take an arbitrary partially-ordered scale $\langle L, \leq \rangle$ of truth degrees and require the existence of infima and suprema (for technical reasons, to be able to evaluate quantifiers). Furthermore, instead of taking one particular fuzzy conjunction $\otimes$ and fuzzy implication $\rightarrow$, we take any $\otimes$ and $\rightarrow$ which satisfy certain conditions. This way, we obtain a structure $\mathbf{L} = \langle L, \leq, \otimes, \rightarrow, \ldots \rangle$ of truth degrees with logical connectives. Although more general than one particular choice of a scale and connectives, such an approach is easier to handle theoretically and supports the symbolical character of our model. Technically speaking, our structure of truth degrees is assumed to be a complete residuated lattice $\mathbf{L} = \langle L, \wedge, \vee, \otimes, \rightarrow, 0, 1 \rangle$, see [12,13] for details.

A favorite choice of $\mathbf{L}$ is $L = [0, 1]$ or a subchain of $[0, 1]$. Examples of pairs of important pairs of adjoint operations are Łukasiewicz ($a \otimes b = \max(a + b - 1, 0)$, $a \rightarrow b = \min(1 - a + b, 1)$), and Gödel ($a \otimes b = \min(a, b)$, $a \rightarrow b = 1$ if $a \leq b$, $a \rightarrow b = b$ else). Note that a special case of a complete residuated lattice is a two-element Boolean algebra of classical (bivalent) logic.

Having $\mathbf{L}$, we define usual notions [12,13,15]: an $\mathbf{L}$-set (fuzzy set) A in universe U is a mapping $A : U \rightarrow L$, $A(u)$ being interpreted as "the degree to which u belongs to A". The operations with $\mathbf{L}$-sets are defined componentwise. Binary $\mathbf{L}$-relations (binary fuzzy relations) between X and Y can be thought of as $\mathbf{L}$-sets in the universe $X \times Y$. A fuzzy relation E in U is called reflexive if for each $u \in U$ we have $E(u, u) = 1$; symmetric if for each $u, v \in U$ we have $E(u, v) = E(v, u)$. A reflexive and symmetric fuzzy relation is called a similarity. We often denote a similarity by $\approx$ and use an infix notation, i.e. we write $(u \approx v)$ instead of $\approx(u, v)$.

2 Relational Algebra and Implementation of Relational Model over Domains with Similarities

2.1 Ranked Tables over Domains with Similarities

We use Y for a set of attributes (attribute names) and denote the attributes by $y, y_1, \ldots$; $\mathbf{L}$ denotes a fixed structure of truth degrees and connectives.

Definition 1. A *ranked data table over domains with similarity relations* (with Y and $\mathbf{L}$) is given by

- *domains*: for each $y \in Y$, D_y is a non-empty set (domain of y, set of values of y);
- *similarities*: for each $y \in Y$, $\approx_y$ is a binary fuzzy relation (called similarity) in D_y (i.e. a mapping $\approx_y$: $D_y \times D_y \to L$) which is reflexive (i.e. $u \approx_y u = 1$) and symmetric ($u \approx_y v = v \approx_y u$);
- *ranking*: for each tuple $t \in \times_{y \in Y} D_y$, there is a degree $\mathcal{D}(t) \in L$ (called rank of t in $\mathcal{D}$) assigned to t.

Remark 1. (1) $\mathcal{D}$ can be seen as a table with rows and columns corresponding to tuples and attributes, like in Tab. 1. Ranked tables with similarities represent a simple concept which extends the concept of a table (relation) of the classical relational model by two features: similarity relations and ranks. In the classical relational model, similarity relations are not present.

(2) $t[y]$ denotes a value from D_y of tuple t on attribute y. We require that there is only a finite number of tuples with non-zero degree. If $L = \{0, 1\}$ and if each $\approx_y$ is ordinary equality, the concept of a ranked data table with similarities coincides with that of a data table over set Y of attributes (relation over a relation scheme Y) of a classical model.

(3) Formally, $\mathcal{D}$ is a fuzzy relation between domains D_y ($y \in Y$). Rank $\mathcal{D}(t)$ is interpreted as a degree to which the tuple t satisfies requirements posed by a query. A table $\mathcal{D}$ representing just stored data, i.e. data prior to querying, has all the ranks equal to 0 or to 1, i.e. $\mathcal{D}(t) = 0$ or $\mathcal{D}(t) = 1$ for each tuple t. Here again, $\mathcal{D}$ can be thought of as a result of a query, namely, a query "show all stored data".

2.2 Relational Algebra

In the classical model, relational algebra is based on the calculus of classical relations. In the same spirit, since ranked tables are in fact fuzzy relations, our relational algebra is based on the calculus of fuzzy relations [12,15]. Due to the limited scope, we present only selected parts of our algebra and leave details and further parts to a full version of the paper.

Table 2. Ranked tables over domains with similarities

$\mathcal{D}_1(t)$	*name*	*age*	*education*
1.0	Black	30	Comput. Eng.
0.9	Chang	28	Accounting
0.1	Francis	39	Business

$\mathcal{D}_2(t)$	*name*	*age*	*education*
1.0	Adams	30	Comput. Sci.
0.8	Davis	27	Comput. Eng.
0.5	Black	30	Comput. Eng.
0.4	Enke	36	Electric. Eng.
0.3	Francis	39	Business

Operations of our relational algebra can be basically classified as follows.

Counterparts to Boolean operations of classical model. These are operations like union, intersection, etc. For instance, a union $\mathcal{D}_1 \cup \mathcal{D}_2$ of two ranked tables with similarities, $\mathcal{D}_1$ and $\mathcal{D}_2$, is defined by

$$[\mathcal{D}_1 \cup \mathcal{D}_2](t) \;=\; \mathcal{D}_1(t) \vee \mathcal{D}_1(t).$$

This says that a rank of t in $\mathcal{D}_1 \cup \mathcal{D}_2$ is given by taking a rank of t in $\mathcal{D}_1$ and a rank of t in $\mathcal{D}_2$, and applying $\vee$ to these ranks. In most situations, $\vee$ coincides with maximum. Therefore, $[\mathcal{D}_1 \cup \mathcal{D}_2](t)$ is just the maximum of $\mathcal{D}_1(t)$ and $\mathcal{D}_2(t)$. For example, the ranked table $\mathcal{D}$ from Tab. 1 is a result of union of ranked tables $\mathcal{D}_1$ and $\mathcal{D}_2$ depicted in Tab. 2, i.e. $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2$.

More generally, for any binary (and similar for other arities) operation $\odot$ with fuzzy relations, we define a corresponding operation (denoted again) $\odot$ which yields for any two ranked tables $\mathcal{D}_1$ and $\mathcal{D}_2$ (with common Y, domains, and similarities) a ranked table $\mathcal{D}$ assigning to any tuple t a rank $\mathcal{D}(t)$ defined componentwise by

$$\mathcal{D}(t) \;=\; \mathcal{D}_1(t) \odot \mathcal{D}_2(t).$$

New operations based on calculus of fuzzy relations. The calculus of fuzzy relations contains operations which either have no counterparts with classical relations or the counterparts are trivial. An interesting example is a so-called a-cut of a fuzzy relation. For a ranked table $\mathcal{D}$ and a rank $a \in L$, an a-cut of $\mathcal{D}$ is a ranked table ${}^a\mathcal{D}$ defined by

$$[{}^a\mathcal{D}](t) = \begin{cases} 1 & \text{if } \mathcal{D}(t) \geq a, \\ 0 & \text{otherwise.} \end{cases}$$

That is, ${}^a\mathcal{D}$ is a non-ranked table which contains those tuples of $\mathcal{D}$ with ranks greater or equal to a. This is quite a natural operation for manipulation of ranked tables which allows the user to select only a part of a query result given by threshold a.

Note that in combination with intersection, we can use a-cut to get the part of $\mathcal{D}$ with ranks at least a, i.e. we can get $\mathrm{Above}_a(\mathcal{D})$ defined by

$$[\mathrm{Above}_a(\mathcal{D})](t) = \begin{cases} \mathcal{D}(t) & \text{for } \mathcal{D}(t) \geq a, \\ 0 & \text{otherwise.} \end{cases}$$

Namely, we have

Table 3. Results of $\text{Above}_{0.7}$ and $\sigma_{e=\text{CE}}$ applied to $\mathcal{D}$ from Tab.1

$[\text{Above}_{0.7}(\mathcal{D})](t)$	*name*	*age*	*education*
1.0	Adams	30	CS
1.0	Black	30	CE
0.9	Chang	28	AC
0.8	Davis	27	CE

$\mathcal{D}(t)$	*name*	*age*	*education*
1.0	Black	30	CE
0.9	Adams	30	CS
0.8	Davis	27	CE
0.1	Enke	36	EE

Lemma 1. *For $\mathcal{D}$, $a \in L$, we have* $\text{Above}_a(\mathcal{D}) = \mathcal{D} \cap {}^a\mathcal{D}$.

For instance, $\text{Above}_{0.7}(\mathcal{D})$ for $\mathcal{D}$ from Tab.1 is depicted in Tab.3 (left).

Counterparts to selection, join, projection, etc. These operations stem basically from the classical ones by taking into account similarity relations (or, in general fuzzy relations θ in place of classical comparators). For illustration, we consider only a similarity-based selection and a similarity-based join.

The basic form of selection works as follows. Given a value $u \in D_y$, a select operator with inputs $y = u$ yields a ranked table $\sigma_{y=u}(\mathcal{D})$ for which a rank of a tuple $t \in \times_{y \in Y} D_y$ is given by

$$\mathcal{D}(t) \otimes (t[y] \approx_y u),$$

i.e.

$$[\sigma_{y=u}(\mathcal{D})](t) \;=\; \mathcal{D}(t) \otimes (t[y] \approx_y u),$$

$[\sigma_{y=u}(\mathcal{D})](t)$ can be read as follows: One takes a degree $\mathcal{D}(t)$ (rank of t in $\mathcal{D}$) and a degree $t[y] \approx_y u$ (degree of similarity between $t[y]$ and u) and applies a "fuzzy conjunction" $\otimes$ to $\mathcal{D}(t)$ and $t[y] \approx_y u$. That is, a rank of t in $\sigma_{y=u}(\mathcal{D})$ can be seen as a truth degree of proposition "t is in $\mathcal{D}$ *and* the value of t in attribute y is similar to u". Tab.3 (right) shows $\sigma_{e=\text{CE}}(\mathcal{D})$ (e denotes "education", CE denotes "Computer Engineering") for $\mathcal{D}$ from Tab.1 for $\otimes$ being Łukasiewicz conjunction.

As in the ordinary case, selection can be extended to several input values. Given $\mathcal{D}$, a select operator with inputs given by $y_1 = u_1, \ldots, y_k = u_k$ yields a ranked table $\sigma_{y_1=u_1,\ldots,y_k=u_k}(\mathcal{D})$ for which a rank of a tuple t is given by

$$\mathcal{D}(t) \otimes (x[y_1] \approx_{y_1} u_1) \otimes \cdots \otimes (x[y_k] \approx_{y_k} u_k),$$

i.e. a degree of "t is in $\mathcal{D}$ and value of t in y_1 is similar to u_1 and $\cdots$ and value of t in y_k is similar to u_k".

To illustrate similarity-based join, consider a ranked table $\mathcal{D}_1$ from Tab.1 which can be thought of as a result to a query "select candidates with age about 30") and a ranked table $\mathcal{D}_2$ from Tab.4 (left) describing open positions with required education. A similarity-based join $\mathcal{D}_1 \bowtie \mathcal{D}_2$ then describes possible job assignments. A rank $[\mathcal{D}_1 \bowtie \mathcal{D}_2](n, a, e, p)$ of tuple $\langle n, a, e, p \rangle$ in $\mathcal{D}_1 \bowtie \mathcal{D}_2$ is given by

$$\bigvee\nolimits_{e_1, e_2} (\mathcal{D}_1(n, a, e_1) \otimes (e_1 \approx_e e) \otimes (e \approx_e e_2) \otimes \mathcal{D}_2(p, e_2))$$

Table 4. Illustration of similarity-based join

$\mathcal{D}(t)$	*position*	*education*
1.0	programmer	Comput. Sci.
1.0	syst. technician	Comput. Eng.

$\mathcal{D}(t)$	*name*	*position*
1.0	Adams	programmer
1.0	Black	syst. technician
0.9	Adams	syst. technician
0.9	Black	programmer

where e_1, e_2 range over the domain corresponding to *education*. That is, the join runs not only over equal values but also over similar values. $[\mathcal{D}_1 \bowtie \mathcal{D}_2](n, a, e, p)$ can be seen as a truth degree of "candidate with name n, age a, and education e_1 belongs to table $\mathcal{D}_1$ and e_1 is similar to e_2 and e_2 is a required education for position p". The table of Tab.4 (right) shows a result of $\mathrm{Above}_{0.9}(\mathcal{D}_1 \bowtie \mathcal{D}_2)$, cf. above, projected to *name* and *position*.

Further operations. Among others, here belong some interesting operations studied in databases and information retrieval. As an example, consider top_k which gained a considerable interest recently, see [10,11] and also [14]. We define $\mathrm{top}_k(\mathcal{D})$ to contain the first k tuples (according to rank ordering) of $\mathcal{D}$ with their ranks (if there are less than k ranks in $\mathcal{D}$ then $\mathrm{top}_k(\mathcal{D}) = \mathcal{D}$; and $\mathrm{top}_k(\mathcal{D})$) includes also the tuples with rank equal to the rank of the k-th tuple). Note that top_k is a part of a query language described in [19]. As presented in [4], top_k is indeed a relational operator, i.e. there is a corresponding formula in the corresponding relational calculus which is based on first-order fuzzy logic.

Remark 2. In [4], we presented a tuple and domain relational calculi for our relational algebra with the completeness theorem.

We obtained several results on properties of the operations of our relational algebra which are analogous to properties of classical relational algebra. Due to the limited scope, we present just the following properties of selection:

Lemma 2. *For $\mathcal{D}$ and $u_i \in D_{y_i}$ we have*

$$\sigma_{y_1=u_1,\ldots,y_k=u_k}(\mathcal{D}) = \sigma_{y_1=u_1}(\cdots(\sigma_{y_k=u_k}(\mathcal{D}))\cdots),$$
$$\sigma_{y_1=u_1}(\sigma_{y_2=u_2}(\mathcal{D})) = \sigma_{y_2=u_2}(\sigma_{y_1=u_1}(\mathcal{D})).$$

2.3 Implementation Issues

This section presents, by means of examples, considerations on and proposal of implementation of the extended relational model. Our basic aim to make use of existing relational database management systems (RDBMS) and to develop a prototype implementation of the extension by ranks and similarities. We used a RDBMS Oracle9i, query language SQL, and its procedural extension PL/SQL. Oracle9i enables us to use stored procedures and functions. This feature enables us to store with a database scheme some functions that can be used in SQL queries and other SQL statements.

We did use stored functions to implement similarity relations on particular domains. Recall that a similarity relation on a domain D_y is, in fact, a function of the Cartesian product $D_y \times D_y$ into a scale L of truth degrees, e.g. into $[0, 1]$. For example, a function `sim_age` implementing similarity $\approx_{age}$ on the domain of ages (see Table 1) can be defined by

```
CREATE FUNCTION sim_age(age1 NUMBER, age2 NUMBER)
  RETURN NUMBER IS
x NUMBER;
BEGIN
  x := 1 - abs(age1-age2)/SCON;
  IF x < 0 THEN
     x := 0;
  END IF;
  RETURN NUMBER(x);
END sim_age;
```

where `SCON` is an appropriate scaling constant. Similarities on non-numerical domains D_y can be implemented as relations over relation scheme given by three attributes: y (first), y (second), and an attribute representing truth degrees (third). For instance, if we consider the ranked data table from Tab. 1, the information about the similarity on the domain of education can be stored in a data base table which is created by the following SQL command:

```
CREATE TABLE sim_education_tab (
  val1 VARCHAR (30) NOT NULL,
  val2 VARCHAR (30) NOT NULL,
  degree NUMBER NOT NULL,
  PRIMARY KEY (val1, val2),
  CHECK (val1 < val2)
);
```

Notice that `val1` and `val2` are string attributes representing education descriptions and `degree` is their similarity degree. Since similarity relations are reflexive, there is no need to store information about similarities when `val1` and `val2` agree on their values. Furthermore, similarity relations are always symmetric. Hence, if `sim_education_tab` contains information about the similarity degree for `val1` $= e_1$ and `val2` $= e_2$, there is no need to store the information for `val1` $= e_2$ and `val2` $= e_1$. Since the domain of education descriptions, which are encoded by strings of literals, can be lexically ordered, we can store in `sim_education_tab` only records representing similarities of education values `val1` $= e_1$ and `val2` $= e_2$ such that e_1 is lexically smaller than e_2 and $e_1 \approx_e e_2 > 0$. The latter SQL command creates `sim_education_tab` with explicit constraint saying that the value of `val1` should be lexically smaller than the value of `val2` which reflects the organization of the data table just mentioned. The table `sim_education_tab` has a primary key $\{$`val1`, `val2`$\}$. In most RDBMS the definition of such a primary key is accompanied by creation of

a unique multi-column index which can significantly improve the efficiency of querying `sim_education_tab` regarding the similarity degree. Following the example from Tab.1, `sim_education_tab` can be filled as follows:

```
INSERT INTO sim_education_tab VALUES ('A', 'B', 0.7);
INSERT INTO sim_education_tab VALUES ('CE', 'CS', 0.9);
INSERT INTO sim_education_tab VALUES ('CE', 'EE', 0.7);
INSERT INTO sim_education_tab VALUES ('CS', 'EE', 0.6);
```

In order to achieve flexibility, we should create a stored function `sim_education` (its source code is not shown here) which given two education descriptions returns their similarity degree, querying the `sim_education_tab` table. Using functions implementing similarities on domains and using standard features of SQL, one can implement queries corresponding to the operations of the relational algebra for ranked tables over domains with similarities.

The similarity-based queries are intended to query ranked tables, i.e. the arguments of similarity-based queries are, in principle, ranked tables. In a conventional RDBMS, a ranked table over relation scheme Y can be represented as a relation over relation scheme Y to which we add an attribute for ranks (first column). However, from the users' point of view, it is convenient to have the option to apply similarity-based queries to ordinary relations (i.e., tables without ranks) as well. Suppose that we have in our RDBMS a relation (called `candidates`) depicted in Tab.1 (without ranks) and want to see the candidates with age similar to 30. This can be accomplished by SQL statement

```
SELECT sim_age(age, 30) AS sim_age_30, *
  FROM candidates
  ORDER BY sim_age_30 DESC;
```

The result is a relation with the first column representing ranks. In terms of our relational algebra, this ranked table is a result of $\sigma_{age=30}(\mathcal{D})$ where $\mathcal{D}$ is the table from Tab.1. In a similar way, using a function implementing fuzzy logical conjunction, one can form SQL statements implementing queries like "select candidates with age around 30 and education similar to electrical engineering". For instance, the standard Łukasiewicz conjunction can be represented by the following stored function:

```
CREATE FUNCTION luk_conj(degree1 NUMBER, degree2 NUMBER)
  RETURN NUMBER IS
x NUMBER;
BEGIN
  x := degree1 + degree2 - 1;
  IF x < 0 THEN
     x := 0;
  END IF;
  RETURN NUMBER(x);
END luk_conj;
```

Using `luk_conj` together with `sim_age` and `sim_education`, we can formulate the above-mentioned similarity-based query as follows:

```
SELECT luk_conj(sim_age(age, 30),
                sim_education(education, 'EE')) AS sim, *
  FROM candidates
  ORDER BY sim DESC;
```

Remark 3. Note that we have not discussed here a user-friendly syntax of an extension of standard SQL which would enable the user to express similarity-based queries in a comfortable form. Such an extension can be accomplished, e.g., by devising a preprocessor which would translate statements of the SQL extension to statements of ordinary SQL. Due to the limited scope, we do not deal with this issue in the present paper. Our intention was to demonstrate that the similarity-based queries can be implemented by means of the stored functions for similarities and by means of the standard SQL.

3 Future Research

Main topics for future research are:

- development of relational algebra with focus on new (non-standard) operations and features;
- development of prototype implementation of the extended Codd's model by means of existing relational data base management systems;
- design of an extension of standard SQL for the extended Codd's model and its implementation (some proposals for "fuzzy SQL" can be found in the literature but our relational model of data is different from the respective models in the literature);
- development of standard issues from relational databases in our extended setting (e.g., data dependencies, redundancy, normalization, and design of databases, optimization issues, etc.).

References

1. Abiteboul, S., et al.: The Lowell database research self-assessment. Comm. ACM 48(5), 111–118 (2005)
2. Belohlavek, R., Vychodil, V.: Functional dependencies of data tables over domains with similarity relations. In: IICAI 2005, Pune, India, December 2005, pp. 2486–2504 (2005)
3. Belohlavek, R., Vychodil, V.: Data tables with similarity relations: functional dependencies, complete rules and non-redundant bases. In: Lee, M.L., Tan, K.-L., Wuwongse, V. (eds.) DASFAA 2006. LNCS, vol. 3882, pp. 644–658. Springer, Heidelberg (2006)
4. Belohlavek, R., Vychodil, V.: Relational model of data over domains with similarities: an extension for similarity queries and knowledge extraction. In: IEEE IRI 2006, The 2006 IEEE Int. Conf. Information Reuse and Integration, Waikoloa Village, Hawaii, USA, September 16–18, 2006, pp. 207–213 (2006)

5. Belohlavek, R., Vychodil, V.: Codd's relational model of data and fuzzy logic: comparisons, observations, and some new results. In: Proc. CIMCA 2006, Sydney, Australia, p. 6 (2006) ISBN 0–7695–2731–0
6. Bosc, P., Kraft, D., Petry, F.: Fuzzy sets in database and information systems: status and opportunities. Fuzzy Sets and Syst. 156, 418–426 (2005)
7. Buckles, B.P., Petry, F.: A fuzzy representation of data for relational databases. Fuzzy Sets Syst. 7, 213–226 (1982)
8. Buckles, B.P., Petry, F.E.: Fuzzy databases in the new era. In: ACM SAC 1995, pp. 497–502, Nashville, TN (1995)
9. Date, C.J.: Database Relational Model: A Retrospective Review and Analysis. Addison-Wesley, Reading (2000)
10. Fagin, R.: Combining fuzzy information from multiple systems. J. Comput. System Sci. 58, 83–99 (1999)
11. Fagin, R.: Combining fuzzy information: an overview. ACM SIGMOD Record 31(2), 109–118 (2002)
12. Gottwald., S.: A Treatise on Many-Valued Logics. Research Studies Press, Beldock, England (2001)
13. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer, Dordrecht (1998)
14. Illyas, I.F., Aref, W.G., Elmagarmid, A.K.: Supporting top-k join queries in relational databases. The VLDB Journal 13, 207–221 (2004)
15. Klir, G.J., Yuan, B.: Fuzzy Sets and Fuzzy Logic. Theory and Applications. Prentice-Hall, Englewood Cliffs (1995)
16. Li, C., Chang, K.C.-C., Ilyas, I.F., Song, S.: RanSQL: Query Algebra and Optimization for Relational top-k queries. In: ACM SIGMOD 2005, pp. 131–142 (2005)
17. Maier, D.: The Theory of Relational Databases. Computer Science Press, Rockville (1983)
18. Medina, J.M., Vila, M.A., Cubero, J.C., Pons, O.: Towards the implementation of a generalized fuzzy relational database model. Fuzzy Sets and Systems 75, 273–289 (1995)
19. Penzo, W.: Rewriting rules to permeate complex similarity and fuzzy queries within a relational dastabase system. IEEE Trans. Knowledge and Data Eng. 17, 255–270 (2005)
20. Prade, H., Testemale, C.: Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries. Inf. Sci. 34, 115–143 (1984)
21. Raju, K.V.S.V.N., Majumdar, A.K.: Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems. ACM Trans. Database Systems 13(2), 129–166 (1988)
22. Takahashi, Y.: Fuzzy database query languages and their relational completeness theorem. IEEE Trans. Knowledge and Data Engineering 5, 122–125 (1993)
23. Zadeh, L.A.: Similarity relations and fuzzy orderings. Information Sciences 3, 177–200 (1971)

A New Way to Aggregate Preferences: Application to Eurovision Song Contests

Jérémy Besson[1,2] and Céline Robardet[1]

[1] INSA-Lyon, LIRIS CNRS UMR5205, F-69621 Villeurbanne cedex, France
[2] UMR INRA/INSERM 1235, F-69372 Lyon cedex 08, France
{Firstname.Name}@insa-lyon.fr

Abstract. Voting systems have a great impact on the results of contests or elections. Simple methods are actually used, whereas they do not provide most accurate results. For example, in the Eurovision Song Contest, the winner may not be the most preferred candidate. Condorcet criterion, which consists in preserving most of the individual votes in the final ranking, seems intuitively the most relevant. In this paper, we propose a new ranking method founded on Condorcet voting count principle which minimizes the number of pairwise inversions of the individual preferences. We propose a two-step method: computing the cycles among vote preferences and removing a minimal set of pairwise preferences to erase all the cycles and turn the votes into a partial order as close as possible to a total order. Finally, we evaluate the impact of our ranking procedure on the last 30 Eurovision Song Contests.

1 Introduction

Combining individual preferences into a collective preference is an important problem with many real life applications where several agents have to take cooperative decisions. It has been extensively studied in the social choice theory field by philosophers that attempt to define most fare voting systems. Whereas the goal of such systems is to ensure that at least half of the voters should get the outcome they want, such a result is not guarantee by existing systems when there are three or more candidates. On the other hand, the wide spread relative majority voting system in which the candidate who receives most of the votes wins, is known to be outperformed by ranked voting systems, in which voters rank candidates in order of preferences. Thus, many sport competitions, but also the popular "Eurovision Song Contest", use ranked voting methods. An other use of these methods concerns metasearch engines that bring a renewed of interest for these methods [1,6].

Borda count, the simplest ranked voting system, is the most widespread method. Unfortunately, such voting procedure can lead to the election of a candidate which is not the one preferred by the majority of voters. On the other hand, Condorcet methods aim at providing a ranking that preserves the majority of opinions. It considers the relative majority between each pair of candidates and elects the candidate who defeats any other ones in pairwise comparisons.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 152–162, 2007.

But such a winner may not exist if it exists majorities of preferences that are not transitive. In the graph representation of the preferences, the intransitivity among preferences is equivalent to the existence of cycles in the graph.

In this paper, we propose a new evaluation procedure. It eliminates cycles among pairwise preferences to obtain a partial order such that the number of eliminated votes is minimized. Computing the optimal partial order is feasible by enumerating potential solutions in such a way that the objective function increases. The current best solution is used to prune branches having a higher objective function value. We also constrain our method to obtain a partial order that is not too far from a total order. A measure based on the number of comparable candidates is used to guarantee that the computed partial order is almost a total order. We evaluate the impact of the ranking procedure on the last 30 Eurovision Song Contests and compare the obtained rankings with the official ones.

This paper is organized as follows. In section 2, we present the background knowledge on ranking vote methods. In section 3, we formalize the problem of aggregating preferences into an optimization problem and propose an algorithm that computes an optimal solution. An evaluation of the proposed method is given in section 4. Official rankings of the last Eurovision Song Contests are compared by the ones obtained using the proposed method. Section 5 concludes.

2 Background

Ranking algorithms specify how to aggregate several rankings into a "consensus" one. Voters order a set of candidates from most to least preferred and thus, a more or less complicated algorithm is applied to aggregate them. Two opposite approaches have mainly been used for more than two centuries: the voting count theory of Borda and the one of Condorcet.

The Borda counting vote principle aggregates preferences by assigning a number of marks to each position in the rank. The candidates are ordered according to the sum of the marks they were granted. Borda counting system is often described as a consensus-based electoral system, rather than a majority-based one because it can lead to the election of a candidate which is not the one preferred by the majority of voters.

In the Condorcet approach, the vote counting consists in simulating all possible duels between candidates: for each pair of candidates, the number of ballot papers on which the first candidate is preferred to the second is counted. Thus the relative majority between each couple of candidates is considered. For each duel, there is a victorious candidate. If one candidate defeats all others in such duels, thus he/she is elected winner. A particular point of interest is that the overall winner might not be the first preference of any voter. In a sense, the Condorcet method yields the "best compromise" candidate, the one that the largest majority will find to be least disagreeable, even if not their favorite.

But such a winner does not exist if the majority preferences are not transitive. Such a situation is called "paradox of Condorcet": it is possible that a majority prefers A to B, and also that another majority prefers B to C, and that a third majority prefers C to A. Thus, it is impossible to rank candidates in a total order only using the majority criterion. Condorcet gave indications on how to break cycles that might occur. However, his prescription is not completely clear. Young [8] shows that a correct application of Condorcet maximum likelihood approach leads to an order that has the maximum pairwise support from the voters. Montague et al.[4] propose to order candidates by partitioning candidates into strongly connected components that correspond to different voting cycles. The components are thus ordered using the Condorcet rule. Candidates belonging to the same component are ranked arbitrarily.

The Kemeny method consists in reversing the preferences that do not satisfy the majority criterion and that have the least combined plurality [7]. Such an order involves the minimum number of pairwise inversions with the individual rankings. Dwork et al. [1] show that finding a Kemeny optimal ranking is an NP-hard problem when the number of voters is even and greater than 4. Zhu et al. propose in [9] an heuristic to approximate the solution of this problem by iteratively incrementing or decrementing the position in the consensus order of the candidate i who has the largest difference between the sum of votes that support i against another candidate and the sum of votes that support others candidates against i.

3 Problem Setting and Contribution

Here, we give our formal model of voting. Given a set of n candidates $\mathcal{C} = \{C_1, \cdots, C_n\}$ and a set of m voters $V = \{V_1, \cdots, V_m\}$, the ranking $<_{V_i}$ of the voter V_i with respect to $\mathcal{C}$ is a partial order on the elements of $\mathcal{C}$ (see Definition 1) and thus can be represented by a directed acyclic graph. $<_{V_i}$ represents the preferences of the voter V_i, that is to say $C_1 \leq_i C_2$ means that voter V_i prefers C_2 to C_1. In the Eurovision Song Contest, each judge gives its top-10 candidates. This ranking is a total order on these 10 candidates. The other candidates have a lower rank and are incomparable to each others. Thus, each voter gives a partial order on $\mathcal{C}$. As in the Condorcet approach, we consider each partial order $<_{V_i}$ as a set of pairwise comparisons.

We have a collection of m rankings $\{<_{V_1}, \cdots, <_{V_m}\}$ which are respectively provided by the element of V. We represent the m rankings by a weighted directed graph $G_m = (\mathcal{C}, \mathcal{A}, \omega)$ where the set of vertices $\mathcal{C}$ is the set of candidates, the set of arcs A is defined by

$$\mathcal{A} = \{(C_i, C_j) \mid C_i, C_j \in \mathcal{C} \textit{ and } \exists V_k \in V \textit{ such that } C_i <_{V_k} C_j\}$$

and ω is the weight function from $\mathcal{A}$ to $\mathbb{N}$ which associates to each arc (C_i, C_j) the number of voters that prefer C_j to C_i:

$$\omega(C_i, C_j) = |\{k \in V \mid C_i <_{V_k} C_j\}|$$

Definition 1. *A strict order $<$ on a set $\mathcal{C}$ is a transitive, irreflexive and anti-symmetric relation. Such an order can be total, if for each pair of elements $(X, Y) \in \mathcal{C}^2$, either we have $X < Y$ or $Y < X$. Otherwise, the order is said to be partial. Two candidates X and Y are said to be incomparable if neither $X < Y$ nor $Y < X$. A partial order can be represented by a DAG (Directed Acyclic Graph) where each vertex represents an element of $\mathcal{C}$ and an arc from X to Y exists if $X < Y$. If the order is total, the graph is a chain.*

A voting algorithm consists in aggregating the m rankings into a "consensus" ranking G_O. G_O is thus a DAG on $\mathcal{C}$. Computing G_O in a Condorcet manner requires the elimination of cycles of G_m. As a consequence some voters' pairwise preferences are not taken into account in G_O. The most relevant consensus order has to minimize the number of eliminated votes. However, taking only this criterion might lead to a partial order that contains too few comparable candidates. In that case, one could prefer to obtain a worse order in the sense of the number of eliminated votes, but with more comparable candidates.

To summarize, we want to compute an order G_O which optimizes the two following criteria:

1. minimizing the number of votes of G_m in conflict with G_O
2. maximizing the number of comparable candidates in G_O

If we consider the whole graph G_m that contains all the pairwise comparisons given by the voters, we may not have an order, i.e. the resulting relation may not be anti-symmetric and G_m might contain cycles. An intuitive algorithm could be to remove a minimal number of pairwise comparisons among those given by the voters to turn the relation into a partial order. In that way, we obtain a partial order on candidates, and by minimizing the number of removed pairwise comparisons, most of the candidates are still comparable to each others. Let us define more formally the used criteria. Given a graph G, $D_1(G)$ evaluates how the candidates are comparable to each others. Here we only consider the structure of the graph G and not its associated weights. D_1 is defined as follows:

$$D_1(G = (\mathcal{C}, \mathcal{A}, \omega)) = \frac{\sum_{k=0}^{n} |\{C_i \in \mathcal{C} \mid (C_i, C_k) \in \mathcal{A} \text{ or } (C_k, C_i) \in \mathcal{A}\}|}{\binom{2}{n}}$$

D_1 computes the number of couples of candidates that are comparable. D_1 is equal to 1 when all the candidates are comparable.

We use another metric $D_2(G)$ to evaluate how many pairwise comparisons of G_m are not included into G. We have:

$$D_2(G = (\mathcal{C}, \mathcal{A}, \omega)) = \sum_{k=0}^{n} |\{(C_i, C_j) \in \mathcal{C}^2 \mid C_i <_{V_k} C_j \text{ and } (C_i, C_j) \notin \mathcal{A}\}|$$

Let us consider the following specialization relation $\subseteq$ on graphs:

Definition 2 (Specialization relation). *Let* $G_1 = (\mathcal{C}, A_1, \omega)$ *and* $G_2 = (\mathcal{C}, A_2, \omega)$ *be two graphs that differ by their set of arcs.*

$$G_2 \subseteq G_1 \Leftrightarrow ((X,Y) \in A_2 \Rightarrow (X,Y) \in A_1)$$

The ideal consensus graph $G_O \subseteq G_m$ is a graph that maximizes D_1 and minimizes D_2. As these two criteria are in a way contradictory, we are looking for a graph with a high value for $D_1(G_O)$ and a small value for $D_2(G_O)$. Formally, the problem is set as follows. Given G_m and a numerical threshold α, the problem consists in computing the graph $G_O \subseteq G_m$ such that:

$$\begin{array}{l} G_O \text{ is a DAG} \\ D_1(G_O) \geq \alpha \\ \textit{minimizing } D_2(G_O) \end{array}$$

The first condition implies that there is no cycle in the graph G_O. The second one ensures that at least α% of the candidates are comparable to each others. Finally, among the graphs that satisfy these two constraints, G_0 is one of the graphs that minimizes the number of deleted votes.

This combinatorial problem requires to optimize a function under contraints. Resolving such a problem necessitates to explor the search space of graphs $G \subseteq G_m$ to find the solution. In a data mining framework, we can take advantages from using an appropriate enumeration process that transforms the constraints and the objective function into monotonic functions that can be actively used to prune the search space. Let us first recall the monotonic constraint definition:

Definition 3. *(Monotonicity) A constraint* $\mathcal{C}$ *is said anti-monotonic w.r.t.* $\subseteq$ *iff* $\forall G_1, G_2$ *two graphs such that* $G_1 \subseteq G_2$, $\mathcal{C}(G_2) \Rightarrow \mathcal{C}(G_1)$. $\mathcal{C}$ *is said monotonic w.r.t.* $\subseteq$ *iff* $\forall G_1, G_2$ *such that* $G_1 \subseteq G_2$, $\mathcal{C}(G_1) \Rightarrow \mathcal{C}(G_2)$.

One can adopt one of these two strategies to explore the search space of graphs $G \subseteq G_m$:

- starting with an empty graph and adding recursively arcs from G_m
- or, starting from G_m and removing arcs

The second enumeration process has the advantage that functions D_1 and D_2 turn to be monotonic. The monotonicity of D_1 and D_2 implies that if it exists a graph G_2 such that $D_1(G_2) < \alpha$ then, $\forall G_1 \subseteq G_2$, $D_1(G_1) < \alpha$. Similarly, if G_2 does not minimize D_2, then $\forall G_1 \subseteq G_2$, G_1 does not minimize D_2 as well. Such conditions can be used to prune the search space. Consequently, due to these properties, our algorithm is based on the second enumeration. The only one constraint which is not automatically satisfied by the enumeration is the DAG constraint which is not monotonic. This constraint is thus ensured by construction: at least one arc by cycle is removed.

Our algorithm proceeds in two phases: it computes all the cycles of G_m and then generates potentially all, the partial orders by removing from G_m at least one arc of each cycle.

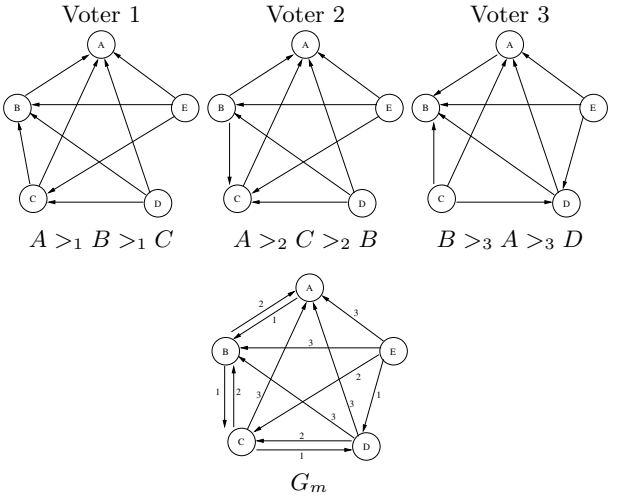


Fig. 1. Graph representation of three voters' rankings that order three candidats among five

We propose a Branch-and-Bound algorithm (see Table 1) to find the partial order G_O derived from G_m having the lowest $D_2(G_O)$ such that $D_1(G_O) > \alpha$. It is a depth-first search algorithm. Let $\mathcal{D}$ be the set of cycles of G and $\mathcal{A}$ the set of arcs belonging to at least one cycle of $\mathcal{D}$. We enumerate all the sub-graphs of G_m such that 1) $D_2(G)$ is lower than the already extracted partial orders and 2) $D_1(G)$ is greater than α.

We select an arc $e_i \in \mathcal{A}$ and we compute $\mathcal{D}'$ by removing from $\mathcal{D}$ the cycles which contain e_i. e_i is only considered if it enables the deletion of cycles from $\mathcal{D}$. Then we call recursively the function CUT() with $\mathcal{A} \setminus \{e_i\}$ and $\mathcal{D}'$ as parameters to compute all the maximal partial orders containing e_i. We also call recursively CUT() with $\mathcal{A} \setminus \{e_i\}$ and $\mathcal{D}$ to compute all the maximal partial orders which do not contain e_i. The enumeration process is stopped when $\mathcal{D}$ is empty. A relevant partial order is thus obtained and stored. Before splitting the search space into two new ones we compute the metric D_1 and D_2 on the current candidate. If $D_1(G) > \alpha$ and $D_2(G)$ is lower than the best solution already computed, then the enumeration process keep going, otherwise it is not split.

Figure 1 represents a ranking problem example on which we explain in the following how the proposed algorithm computes a consensus order. Three voters have ranked three candidates among five denoted by A, B, C, D *and* E. The first voter chooses the order $A \geq_1 B \geq_1 C$, the second one $A \geq_2 C \geq_2 B$ and the last one $B \geq_3 A \geq_3 D$. The graph representing the three voters' rankings is shown on the top of Figure 1. The graph G_m, obtained by merging the previous graphs, is reported on the bottom of Figure 1.

Figure 2 represents the main steps of the proposed algorithm when it is applied on the votes of Figure 1. First it computes all the cycles of G_m. On these data,

Table 1. Pseudo-code of the proposed algorithm

$G_m = (\mathcal{C}, E, \omega)$ the voting graph, $\mathcal{D}$ a set of cycles, $\mathcal{A}$ a set of arcs, $\mathcal{R}$ the set of removed arcs and MIN the lowest D_1 of the already extracted partial orders.

Algo()

$\mathcal{D} \leftarrow Cycles(G)$
$\mathcal{A} \leftarrow \{(X, Y) \in \mathcal{D}\}$
$Min \leftarrow +\infty$
$Cut(\mathcal{D}, \mathcal{A}, \emptyset)$

End Algo

Cut$(\mathcal{D}, \mathcal{A}, \mathcal{R})$

if($D_1(G = (\mathcal{C}, E \setminus \mathcal{R}, \omega)) > \alpha$ and $D_2(G = (\mathcal{C}, E \setminus \mathcal{R}, \omega)) < Min$)
if($\mathcal{D} \neq \emptyset$)
Let $e_i \in \mathcal{A}$
$\mathcal{D}' \leftarrow \{c \in \mathcal{D} \mid e_i \notin c\}$
$Cut(\mathcal{D}', \mathcal{A} \setminus \{e_i\}, \mathcal{R} \cup \{e_i\})$
$Cut(\mathcal{D}, \mathcal{A} \setminus \{e_i\}, \mathcal{R})$
Else
$Store\ G = (\mathcal{C}, E \setminus \mathcal{R}, \omega)$
$Min \leftarrow D_2(G = (\mathcal{C}, E \setminus \mathcal{R}, \omega))$
End if
End if

End Cut

six cycles have been obtained (see Figure 2 left). The right part of the figure presents the enumeration tree. At this step, $|\mathcal{D}| = 6$ and $Min = +\infty$. An arc among those having the lowest weight is chosen. Here it is $B \rightarrow C$ which has a weight of 1. $B \rightarrow C$ appears in four among the six cycles, consequently at most two additional recursive calls are needed to explore this branch in a depth first search approach. The second recursive call leads to the choice of $A \rightarrow B$, and the last cycle is broken by removing the arc $C \rightarrow D$. In the next recursive call, $\mathcal{D}$ is empty and thus the current solution is stored and the variable Min is updated by the sum of the weights of the three removed arcs. Returning to the previous call of the function CUT(), another arc is chosen. So, instead of $C \rightarrow D$, the arc $D \rightarrow C$ is chosen. This choice leads to a graph having a value of four on D_2 and thus the exploration of this branch is stopped. An optimization procedure, which is not depict in the algorithm of Table 1, is applied here to stop the enumeration process. As none of the arcs of the third cycle can produce an acceptable solution, it is not necessary to keep going the enumeration of this branch. Then the algorithm backtrack and instead of $A \rightarrow B$, it enumerates $B \rightarrow A$. Once again the enumeration is stopped because the current solution has a value equal to Min. Other branches of the enumeration tree are shown on Figure 2. Finally, the first extracted order is one of the optimal solutions.

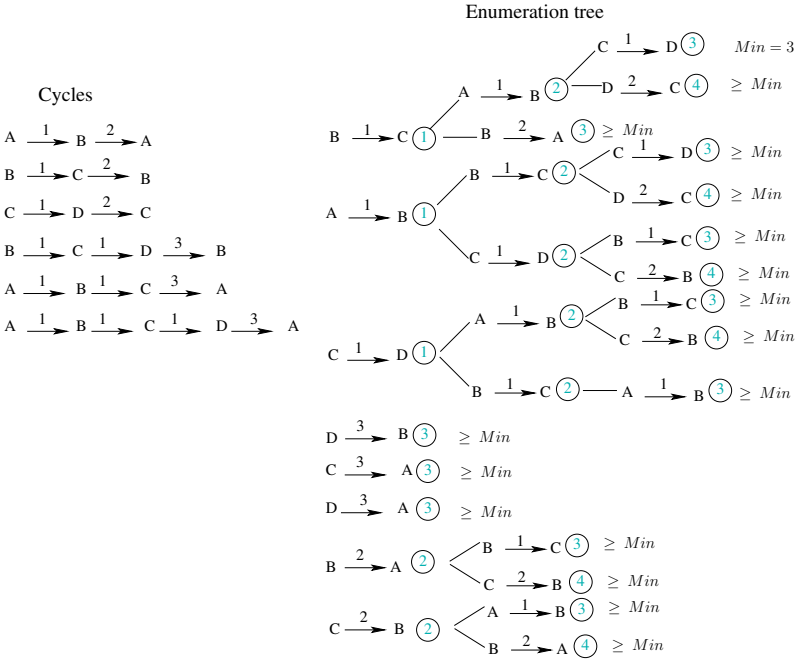


Fig. 2. Application of the proposed algorithm on the votes of Figure 1

4 Experimentation

Eurovision Song Contest data gather participating countries votes to countries' singers during 49 years (between 1957 and 2005). During these years, 43 countries took place in this competition. The voting system used has changed throughout the years and the modern one was set up in 1975. It is an adaptation of the Borda counting vote principle where each voter ranks candidates in order of preferences. The score of a candidate is determined by summing the points corresponding to his/her position in each voter rank. In the Eurovision Song Contest, country judges dispatch a set of marks from 1 to 8, then 10 and finally 12 among competing singers but cannot vote for their compatriot singer. The singer with the best score is the winner of the contest.

The Eurovision Song Contest has been subject to criticisms: judges would vote preferentially for countries independently of the singers' performances. Some analysis of voting patterns [2,3] attest this fact. Figure 3 is an illustration of this point. It represents the distribution of the highest mark (12) given by a country (in abscissa) to the country singers (in stacks). We can see that the distribution of some countries are very biased toward one or a couple of preferential countries. For example, Slovenia gave 60% of its highest mark to Croatia, as Cyprus did for Greece. Estonia gave its highest mark to Sweden in 40% of the contests.

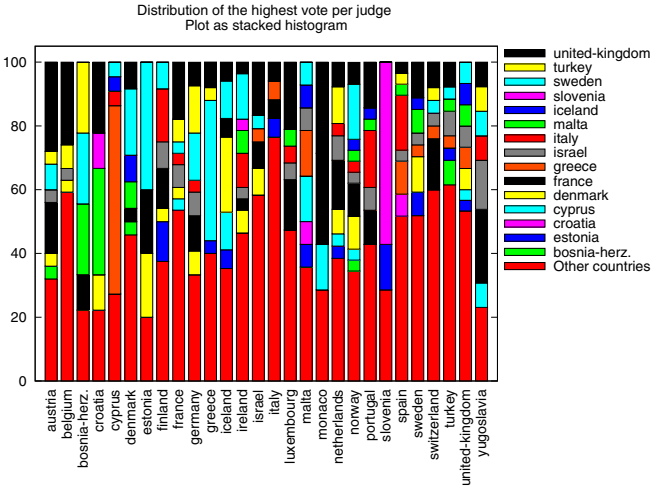


Fig. 3. Number of time that a country (abscissa) gave its highest score to another country (inside the stacks)

We want to test if our ranking method is relevant to determine Eurovision Song Contest winners. For each year between 1975 and 2005 we compute the voting graph that gathers countries' vote preferences. We remove arcs that are not supported by a lot of countries. We fix α to 30%. Due to the lack of space we only report results for year 1975 and 1989.

Figure 4 shows the ranking obtained by the proposed method (black arrows) for the contest held in 1976. The official ranking is also represented with gray arrows which are dotted if they do not belong to the previous order. The proposed method provides a partial order. The top-50% singers are totally ordered, whereas in the ranking tail they are mostly incomparable. As we can see, this computed ranking is very similar to the official one, but the official winner, the United-Kingdom, would be ranked second, behind France[1], by our method.

To assess the quality of the method, we compare the two rankings with an external criterion based on frequent sub-orders. A good ranking should aggregates most of votes' preferences. To make a fair comparison with the Borda method used in the official ranking, we do not used the number of pairwise comparisons conserved by each ranking, because it is the criterion optimized by our method. We choose to compare rankings with respect to the number of sub-orders of length at least 3 that are frequently supported by voters. We use a local pattern extraction method [5] to compute these frequent sub-orders of votes. We extract all sub-orders of length at least 3 supported by at least 4 voters. We obtained 107 sub-orders. 60% of them respect the official ranking whereas 70% of them are consisting with the ranking computed with our method. Thus our ranking method is more coherent with the frequent sub-orders than the official one.

[1] This result is totally independent from the nationality of the authors.

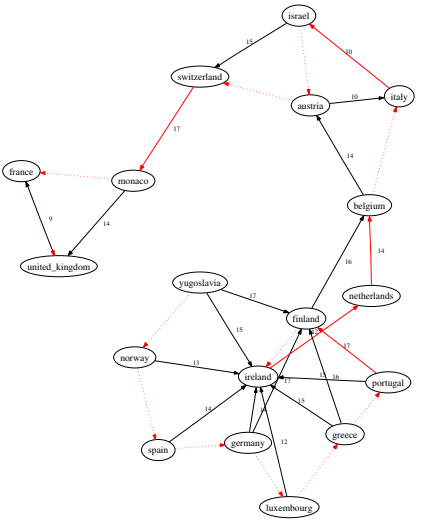


Fig. 4. Representation of the computed ranking (black) and the official one (gray) for the year 1975

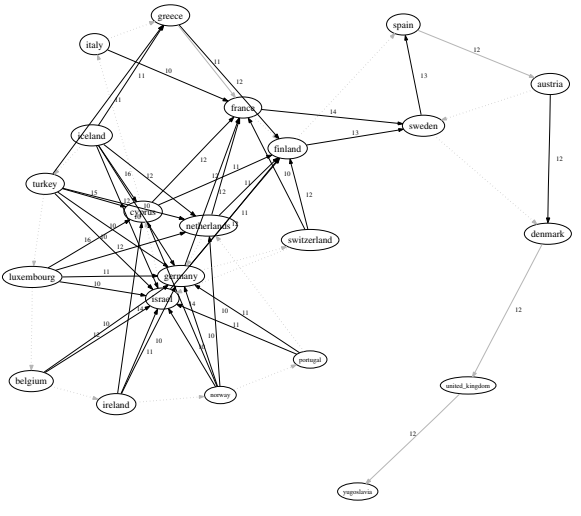


Fig. 5. Representation of the computed ranking (black) and the official one (gray) for the year 1989

Figure 5 presents the obtained ranking for the contest held in 1989. As previously, the top countries are totally ordered in a way similar to the official ranking. There are much more incomparable countries. While computing frequent

sub-orders (of length at least 3 and a frequency greater than 4), we obtained only 15 patterns. This indicates the lack of consensus among voters. Only 30% of them are in the same order than the official ranking, whereas our method conserves 50% of them. When computing the same pattern with a frequency greater than 3, we obtained 91 sub-orders and 50% of them are compatible with the official ranking whereas 60% are sub-orders of the ranking produced by our method.

5 Conclusion

We have introduced a new method to aggregate vote preferences. It preserves most of pairwise preferences embedded in the ranking votes, eliminating only those introducing intransitivity. A parameter is used during computation to make the partial order converge towards a total order. Thus, we have a trade-off between the quality of the computed order and its adequacy with the initial votes. The algorithm minimizes a function which increases during the enumeration. Thus the extraction of the optimal solution is tractable on the Eurovision Song Contest data. The enumeration is aborted when the current solution has a greater value than the current best one. The first results show that the method computes coherent order with the Borda method used in the official ranking. We have shown that the proposed method preserves more frequent sub-orders of the vote rankings and thus better synthesizes votes. We plan to make additional experimentation and especially to further study the impact of parameter α.

Acknowledgements. This research is partially funded by the EU contract IQ FP6-516169 (FET arm of the IST programme).

References

1. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the web. In: WWW10, pp. 613–622 (2001)
2. Fenn, D., Suleman, O., Efstathiou, J., Johnson, N.F.: How does europe make its mind up? connections, cliques, and compatibility between countries in the eurovision song contest. Physica A: Stat. Mechanics and its Applications 360, 576–598 (2006)
3. Ginsburgh, V., Noury, A.: The eurovision song contest is voting political or cultural? (2006), http://www.core.ucl.ac.be/services/psfiles/dp05/dp2005_6.pdf
4. Montague, M., Aslam, J.: Condorcet fusion for improved retrieval. In: CIKM52002, pp. 4–9 (2002)
5. Nanni, M., Rigotti, C.: Quantitative episode trees. In: Proceedings of the 5th international KDID workshp, pp. 95–106 (2006)
6. Renda, E.M., Straccia, U.: Web metasearch: rank vs. score based rank aggregation methods. In: SAC 2003, pp. 841–846. ACM Press, New York (2003)
7. Truchon, M.: An extension of the concordet criterion and kemeny orders. Cahiers de recherche 9813, Université Laval (1998)
8. Young, H.P.: Condorcet's theory of voting. American Political Science Review 82, 1231–1244 (1988)
9. Zhu, S., Fang, Q., Deng, X., Zheng, W.: Metasearch via voting. In: Liu, J., Cheung, Y.-m., Yin, H. (eds.) IDEAL 2003. LNCS, vol. 2690, pp. 734–741. Springer, Heidelberg (2003)

Conditional Classification Trees Using Instrumental Variables

Valerio A. Tutore, Roberta Siciliano, and Massimo Aria

Department of Mathematics and Statistics
University of Naples Federico II, Italy
Via Cintia, M.te Sant'Angelo, 80126 Napoli
{v.tutore,roberta,aria}@unina.it

Abstract. The framework of this paper is supervised learning using classification trees. Two types of variables play a role in the definition of the classification rule, namely a response variable and a set of predictors. The tree classifier is built up by a recursive partitioning of the prediction space such to provide internally homogeneous groups of objects with respect to the response classes. In the following, we consider the role played by an instrumental variable to stratify either the variables or the objects. This yields to introduce a tree-based methodology for conditional classification. Two special cases will be discussed to grow *multiple discriminant trees* and *partial predictability trees*. These approaches use discriminant analysis and predictability measures respectively. Empirical evidence of their usefulness will be shown in real case studies.

1 Introduction

1.1 Nowdays Data Analysis

Understanding complex data structures in large databases is the new challenge for statisticians working in a variety of fields such as biology, finance, marketing, public governance, chemistry and so on. Complexity often refers to both the high dimensionality of units and/or variables and the specific constraints among the variables. One approach is Data Mining [6], namely the science of extracting useful information from large data sets by means of a strategy of analysis considering data preprocessing and statistical methods. Another approach is Machine Learning that combines data-driven procedures with computational intensive methods by exploiting the information technology such to obtain a comprehensive and detailed explanation of the phenomenon under analysis. Turning data into information and then information into knowledge are the main steps of the knowledge discovery process of statistical learning [7] as well as of intelligent data analysis [5]. Key questions in the choice of the best strategy of analysis refer to the type of output (i.e., regression or classification), the type of variables (i.e., numerical and/or categorical), the role played by the variables (i.e., dependent or explanatory), the type of statistical units (i.e., observational or longitudinal data), the type of modelling (i.e., parametric or nonparametric).

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 163–173, 2007.

1.2 Binary Segmentation

In this framework, segmentation methods have proved to be a powerful and effective nonparametric tool for high-dimensional data analysis. A tree-based partitioning algorithm of the predictor space allows to identify homogeneous sub-populations of statistical units with respect to a response variable. A tree path describe the dependence relationship among the variables explaining the posterior classification/prediction of units. Any induction procedure allows to classify/predict new cases of unknown response [13]. In this paper, we refer to CART methodology for classification trees [2] and some advancements provided by two-stage segmentation [8] and the fast algorithm [9]. At each node of the tree, a binary split of sample units is selected such to maximize the decrease of impurity of the response variables when passing from the top node to the two children nodes. This objective function can be shown to be equivalent to maximizing the predictability measure of the splitting variable to explain the response variable. Typically, candidate splits are all possible dichotomous variables that can be generated by all predictor variables, but the fast algorithm allows to find the optimal solution of CART without trying out all possible splits.

1.3 This Paper Genesis

As a matter of fact, when dealing with complex relations among the variables, any CART-based approach offers unstable and not interpretable solutions. An alternative in case of within-groups correlated inputs (of numerical type) is two-stage discriminant trees, where splitting variables are linear combinations of each group of inputs derived by the discriminant factorial analysis [10].

This paper aims to define a segmentation methodology for three-way data matrix starting from some recent results [15]. A three-way data matrix consists of measurements of a response variable, a set of predictors, and in addition a stratifying or descriptor variable (of categorical type). The latter play the role of conditional variable for either the predictor variables or the objects (or statistical units). Two basic methods will be discussed in details providing some justification for the concept of supervised conditional classification in tree-based methodology. Our proposed segmentation methods for complex data structures are all introduced in the Tree-Harvest Software [1] [14] using MATLAB.

2 Multiple Discriminant Trees

2.1 Notation and Definition

Let Y be the output, namely the response variable, and let $\mathbf{X} = \{X_1, \ldots, X_M\}$ be the set of M inputs, namely the predictor variables. In addition, let Z_P be the stratifying predictor variable with G categories. The response variable is a nominal variable with J classes and the M predictors are covariates, thus of numerical type. The input variables are stratified into G groups, on the basis of the instrumental variable Z_P. The g-th block of input variables includes m_g input variables $\mathbf{X}_g = ({}_gX_1, \ldots, {}_gX_{m_g})$ for $g = 1, \ldots, G$.

2.2 The Multiple Method

The proposed method aims to replace blocks of covariates by their linear combinations applying the factorial linear discriminant analysis. In particular, the discriminant analysis is applied twice, first to summarize each block of input variables (within-block latent compromise) and then to find a compromise of all blocks (across-blocks latent compromise). In the first stage, the attention is shifted from the G sets of original covariates to G latent variables, obtained searching for those linear combinations of each block of original variables which summarize the relationships among the covariates with respect to a grouping variable. In our approach, the role of grouping variable is played by the response variable. In the second stage, the algorithm runs to the creation of one global latent variable, synthesis of the previous discriminant functions obtained in the first step. On the basis of such compromise the best split will be found.

2.3 Within-Block Latent Compromises

The process is to divide up a m_g dimensional space into pieces such that the groups (identified by the response variable) are as distinct as possible. Let $\mathbf{B}_g$ be the between group deviation matrix of the inputs in the g-th block and $\mathbf{W}_g$ the (common) within group deviation matrix. The aim is to find the linear combination of the covariates:

$$\phi_g = \sum_{m=1}^{m_g} {}_g\alpha_m \cdot {}_gX_{mg} \tag{1}$$

where ${}_g\alpha_m$ are the values of the eigenvector associated to the largest eigenvalue of the matrix ${\mathbf{W}_g}^{-1}\mathbf{B}_g$. The (1) is the g-th linear combination of the inputs belonging to the g-th block with weights given by the first eigenvector values. It is obtained maximizing the predictability power of the m_g inputs to make the J classes as distinct as possible. Moreover, the ϕ_g variables are all normalized such to have mean equal to zero and variance equal to one.

2.4 Across-Block Latent Compromise

As second step, we find a compromise of the G blocks applying the linear discriminant analysis once again, thus obtaining:

$$\psi = \sum_{g=1}^{G} \beta_g \phi_g \tag{2}$$

where β_g are the values of the eigenvector associated to the the largest eigenvalue of the matrix $\mathbf{W}^{-1}\mathbf{B}$. These matrices refer to the between group deviation matrix of the discriminant functions ϕ_g for $g = 1, \ldots, G$.

2.5 Multiple Factorial Split

Finally, the best split will be selected among all possible dichotomizations of the ψ variable maximizing the decrease of impurity function. As a result, a multiple discriminant split is found where all covariates play a role that can be evaluated considering both the set of coefficients of the linear combination of the blocks and the set of coefficients of the linear combination of the covariates belonging to each block.

2.6 The Computational Steps

The three-steps procedure can be justified with a two-fold consideration: on one hand, a unique global latent and discriminant variable (i.e., the linear combination of within-block latent variables) allows to generate binary splits for that all predictors have contributed; on the other hand, taking into account the within-block latent variables, it is possible to calculate a set of coefficients that represent each the weight of the link among those, the predictors, the response variable and the global latent variable. In other words, if the conditions for the application are verified, the addition of a third stage allows a better interpretation for the explanation of the phenomenon, because all the variables act simultaneously at the same time the split is created, but it is possible to interpret the valence of each of those towards both response and dimensional latent variables.

3 Partial Predictability Trees

3.1 Notation and Definition

Let Y be the output, namely the response variable, and let $\mathbf{X} = \{X_1, \ldots, X_M\}$ be the set of M inputs, namely the predictor variables. In addition, let Z_O be the stratifying object variable with K categories. The response variable is a nominal variable with J classes and the M predictors are all categorical variables (or categorized numerical variables). The sample is stratified according to the K categories of the instrumental variable Z_O.

3.2 The Partial Method

We consider the two-stage splitting criterion [8] based on the predictability τ index of Goodman and Kruskal [3] for two-way cross-classifications: in the first stage, the best predictor is found maximizing the global prediction with respect to the response variable; in the second stage, the best split of the best predictor is found maximizing the local prediction. It can be demonstrated that skipping the first stage maximizing the simple τ index is equivalent to maximizing the decrease of impurity in CART approach. In the following, we extend this criterion in order to consider the predictability power explained by each predictor/split with respect to the response variable conditioned by the instrumental variable

Z_O. For that, we consider the predictability indexes used for three-way cross-classifications, namely the multiple τ_m and the partial τ_p predictability index of Gray and Williams [4], that are extensions of the Goodman and Kruskal τ_s index.

3.3 The Splitting Criterion

At each node, in the first stage, among all available predictors X_m for $m = 1, \ldots, M$, we maximize the partial index $\tau_p(Y|X_m, Z_O)$ to find the best predictor X^* conditioned by the instrumental variable Z_O:

$$\tau_p(Y|X_m, Z_O) = \frac{\tau_m(Y|X_m Z_O) - \tau_s(Y|Z_O)}{1 - \tau_s(Y|Z_O)} \tag{3}$$

where $\tau_m(Y|X_m Z_O)$ and $\tau_s(Y|Z_O)$ are the multiple and the simple predictability measures. In the second stage, we find the best split s^* of the best predictor X^* maximizing the partial index $\tau_s(Y|s, Z_O)$ among all possible splits of the best predictor. It can be possible to apply a CATANOVA testing procedure using the predictability indexes calculated on an indipendent test sample as stopping rule [12].

4 Applications

4.1 Partial Predictability Trees: German Credit Survey

There are several fields in which this methodology can be applied with good results. In this section, we present an application about credit leave in Germany. The data regard a survey collected by Professor Dr. Hans Hofmann, University of Hamburg, with $N = 2026$ [11]. Table 1 describes the predictors of German credit dataset. The response variable is a dummy variable, namely the good and the bad client of the bank.

Table 1. Predictors in the German Credit Dataset

Predictors	
1. Account Status	10. Present residence since
2. Credit history	11. Property
3. Purpose	12. Age
4. Duration	13. Other installment plans
5. Saving accounts/bond	14. Housing
6. Present employment since	15. Existing credits at this bank
7. Installment rate of disposable income	16. Job
8. Personal status and gender	17. People being liable to provide maintenance for
9. Others debitors/guarantors	18. Telephone

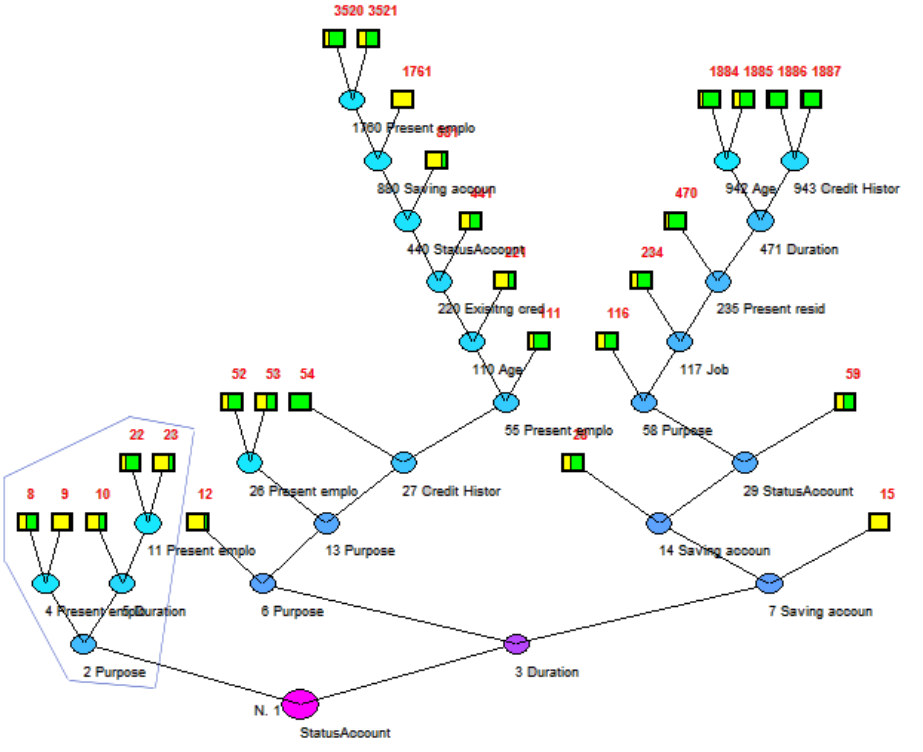


Fig. 1. Partial Predictability Tree Graph (German Credit Survey)

A proper classification rule should consider the different typologies of bank customer. This can be considered as the instrumental Z_O having four strata ordered on the basis of the credit amount requested. Figure 1 shows the final binary tree with 26 terminal nodes, where nodes are numbered using the property that the node t generates the left subnode $2t$ and the right subnode $2t + 1$; we denote the predictor used for the split at each nonterminal node and by distinct color the response class distribution within each terminal node. The branch hold by node 2 is described in details in Figure 2. It is interesting to point out some useful information by interpreting this type of output results. Each node is divided in four parts (one for every category of the instrumental variable) and close to the terminal nodes the percentage of good classified within each group is indicated. In addition, the predictor used for the splits of all strata of cases is also indicated. It can be noticed that the split at the node 2 is based on the credit purpose: in the left subnode (i.e., the node 4) there are relatively more good clients than bad clients as soon as the credit amount increases, their credit is for new car, education and business; in the right subnode (i.e., the node 5) there are relatively more bad clients and the good clients are identified in the further split on the basis of a duration of the credit lower than 12 months.

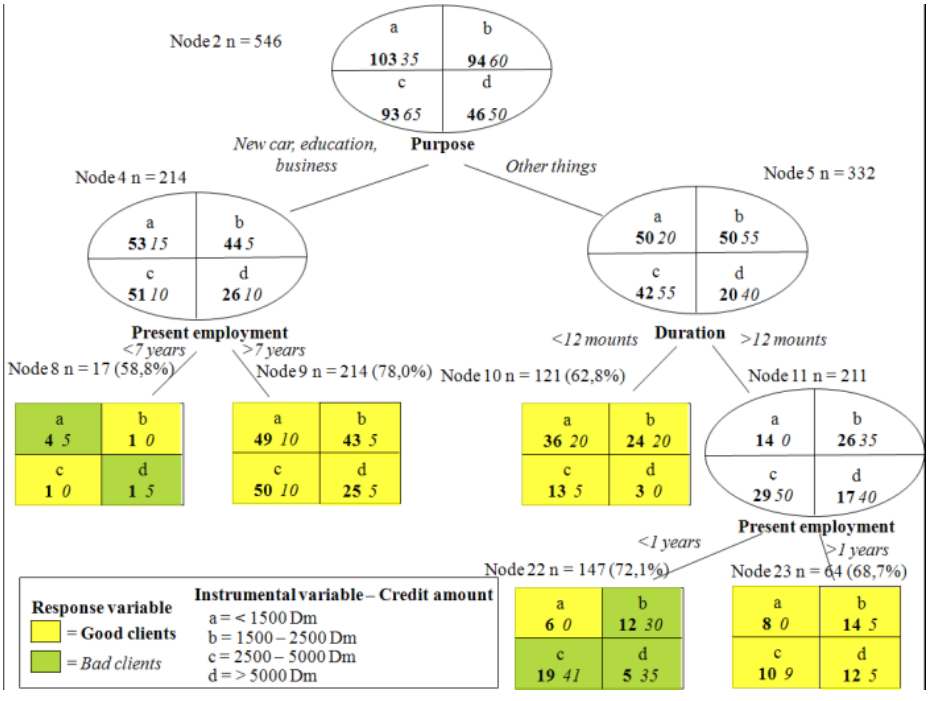


Fig. 2. An example of data interpretation: branch of node 2 (German Credit Survey)

Finally, clients employed by less than 1 year ican be considered as a bad client as soon as the credit amount increases. As further examples of output, Table 2 provides summary information concerning the path yielding to the terminal node 23 which label is good client, whereas Table 3 concerns the path yielding to the terminal node 54 which label is bad client. In particular, in each table we report the response classes distribution of the objects within the four strata of Z_O, for the predictor selected in each nonterminal node. In addition, we give the Catanova test significance value.

As an example, we can see in Table 2 that the original scenario shows a bigger presence of bad clients than good clients in all four strata. After the first split, the situation changes in the first three strata, instead in the fourth there are more bad clients than good. Only after four splits in all strata there is a bigger presence of good clients although there are clear differences within each stratum.

4.2 Multiple Discriminant Trees: Local Transport Survey

Multiple discriminant tree method has been fruitfully applied for a Customer Satisfaction Analysis. On 2006, a survey of $N = 1290$ customers of a local public transport company in Naples has been collected measuring the level of global satisfaction and the level of satisfaction with respect to four dimensions of the service, each considering three aspects.

Table 2. Path 1 – 23: Terminal label - Good client - (German Credit Survey)

Response Classes Distribution			z_1		z_2		z_3		z_4		
Node	**n**	**Predictor**	**G**	**B**	**G**	**B**	**G**	**B**	**G**	**B**	**C sign.**
1	2026	Account Status	205	415	171	295	199	375	91	275	0,0000
2	546	Purpose	103	35	94	60	93	65	46	50	0,0000
5	332	Duration	50	20	50	55	42	55	20	40	0,0000
11	211	Present employm. since	14	0	26	35	29	50	17	40	0,0000
23	64	*Terminal Node*	8	0	14	5	10	10	12	5	0,0080

Table 3. Path 1 – 54: Terminal label - Bad client - (German Credit Survey)

Response Classes Distribution			z_1		z_2		z_3		z_4		
Node	**n**	**Predictor**	**G**	**B**	**G**	**B**	**G**	**B**	**G**	**B**	**C sign.**
1	2026	Account Status	205	415	171	295	199	375	91	275	0,0000
3	1480	Duration	102	380	77	235	106	310	45	225	0,0031
6	720	Purpose	95	295	56	95	44	85	10	40	0,0004
13	698	Purpose	89	290	50	95	39	85	10	40	0,0027
27	491	Credit History	50	220	28	65	28	70	10	20	0,0028
54	71	*Terminal Node*	0	40	0	15	1	10	0	5	0,0195

Table 4. Path 1-8: Terminal label - Unsatisfied customers of Local Transport System

Node	**n**	**score**	**DIM**	1	2	3	4
node 1	1290	145.42	BETA	**0.59**	0.41	0.45	0.34
			ALPHA	10.53	10.50	5.68	9.70
				5.45	4.80	6.01	6.06
				8.80	6.93	10.35	7.30
node 2	473	106.07	BETA	**1.15**	1.01	1.02	0.94
			ALPHA	2.69	1.92	2.00	2.35
				0.41	2.31	1.47	1.31
				2.38	3.40	2.23	2.83
node 4	129	68.27	BETA	1.13	**1.25**	0.98	1.17
			ALPHA	1.73	2.07	0.53	1.00
				0.85	0.31	0.74	1.80
				2.50	2.09	2.97	0.46
node 8	51		*terminal node*				

The response variable has two classes distinguishing the satisfied and the unsatisfied customers. The strata of the instrumental variable Z_P are service's reliability, informations, additional services and travel's comfort, each is characterized by three ordinal predictors, where a Thurstone data transformation has allowed to treat them as numerical ones. Figure 3 describes the role played by the variables in discriminating between satisfied and unsatisfied customers. Table 4 provides summary information concerning the path yielding to a terminal node

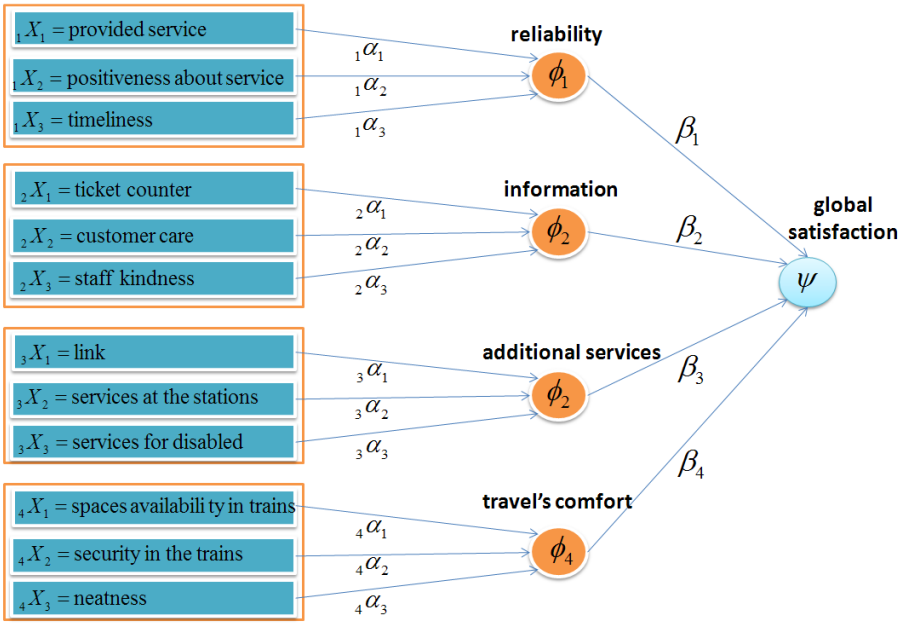


Fig. 3. Discriminant Satisfaction Schema (Local Transport System Survey)

Table 5. Path 1-55: Terminal label - Satisfied customers of Local Transport System

Node	n	score	DIM	1	2	3	4
node 1	1290	145.42	BETA	**0.59**	0.41	0.45	0.34
			ALPHA	10.53	10.50	5.68	9.70
				5.45	4.80	6.01	6.06
				8.80	6.93	10.35	7.30
node 3	817	117.71	BETA	**0.67**	0.24	0.43	0.37
			ALPHA	7.54	7.23	6.68	8.27
				6.85	4.27	3.52	6.07
				7.87	5.75	7.61	5.87
node 6	300	92.44	BETA	**1.01**	0.80	0.89	0.87
			ALPHA	1.19	1.29	1.72	2.56
				2.82	2.25	1.74	1.59
				4.51	0.81	1.66	3.18
node 13	210	46.74	BETA	**0.72**	0.04	0.36	0.28
			ALPHA	4.32	3.25	5.71	3.81
				3.59	1.51	0.00	3.31
				3.54	2.24	3.72	2.96
node 27	122	34.43	BETA	**0.77**	0.14	0.29	0.28
			ALPHA	2.52	1.36	4.31	2.67
				3.01	2.03	0.27	3.17
				3.21	1.52	2.51	1.64
node 55	51		*terminal node*				

which label is satisfied customer, whereas Table 5 concerns the path yielding to a terminal node which label is unsatisfied customer. Table 4 describes for each node the number of individuals, the split-score, the BETA coefficients of each dimension and the ALPHA coefficients within each dimension. From the Table 4 it is clear the high strength of dimension 1 in the split of node 1 and of node 2 as the BETA coefficient is relatively bigger for dimension 1 with respect to the others. Only in the split of node 4 of this path another dimension has the highest coefficient. In the Table 5 for every split the highest BETA coefficient is always in the dimension 1. It is evident that the satisfied customers are well discriminated considering just the dimension relative to reliability; instead for the unsatisfied customers are important all the dimensions of the service.

5 Concluding Remarks

This paper has provided conditional classification trees using an instrumental variable. Two cases have been discussed. In the first, the response variable is a dummy variable, the predictors are numerical and the instrumental variable provides to distinguish them into a set of different blocks. Standard tree-based models would find at each node a split based on just one predictor regardless the multi-block structure of the predictors that can be internally correlated. We have introduced a method to grow *multiple discriminant trees* where the discriminant analysis is used to summarize the information within each block and a multiple splitting criterion has been defined accordingly. At each node of the tree, we are able to assign a coefficient of importance to each predictor within each block as well as to each block in the most suitable discrimination between the two response classes. A Customer Satisfaction Analysis based on a real data set has been briefly presented in order to show some issues in the interpretation of the results. The second case deals with all categorical variables and the instrumental variable provides to distinguish different subsamples of objects. A standard splitting criterion would divide the objects regardless their subsamples belonging. We have introduced a splitting criterion that finds the best split conditioned by the instrumental variable. This yields to grow *partial predictability trees* that can be understood as an extension of two-stage segmentation and to some extent of CART approach. An application on a well-known real data set has been briefly described in order to point out how the procedure gives, at each node, the set of response class distributions, one for each sub-sample. The results of several applications have been very promising for both methods, showing that our methodology works much better than CART standard procedure to explain the interior structure of tree models. Both the two procedures have been implemented in MATLAB environment enriching the Tree Harvest Software we are developing as an alternative to standard tree-based methods for special structures of data.

Acknowledgments. Financial support by MIUR2005 and European FP6 Project iWebCare IST-4-028055. Authors are grateful to useful comments of anonymous referees.

References

1. Aria, M., Siciliano, R.: Learning from Trees: Two-Stage Enhancements. In: CLADAG 2003, Book of Short Papers, Bologna, September 22-24, 2003, pp. 21–24. CLUEB, Bologna (2003)
2. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees, Belmont C.A. Wadsworth (1984)
3. Goodman, L.A., Kruskal, W.H.: Measures of association for cross classifications. Springer, Heidelberg (1979)
4. Gray, L.N., Williams, J.S.: Goodman and Kruskal's tau b: multiple and partial analogs. In: Proceedings of the Americal Statistical Association, pp. 444–448 (1975)
5. Bertold, M., Hand, D. (eds.): Intelligent Data Analysis, 2nd edn. Springer, New York, LLC (2003)
6. Hand, D.J., Mannila, H., Smyth, P.: Principles of Data Mining. The MIT Press, Cambridge (2001)
7. Hastie, T.J., Tibshirani, R.J., Friedman, J.: The Elements of Statistical Learning. Springer, Heidelberg (2001)
8. Mola, F., Siciliano, R.: A two-stage predictive splitting algorithm in binary segmentation. In: Dodge, Y., Whittaker, J. (eds.) Computational Statistics: COMPSTAT '92, pp. 179–184. Physica Verlag, Heidelberg (D) (1992)
9. Mola, F., Siciliano, R.: A Fast Splitting Procedure for Classification Thees. Statistics and Computing 7, 208–216 (1997)
10. Mola, F., Siciliano, R.: Discriminant Analysis and Factorial Multiple Splits in Recursive Partitioning for Data Mining. In: Roli, F., Kittler, J. (eds.) Proceedings of International Conference on Multiple Classifier Systems, Chia, June 24-26, 2002. LNCS, pp. 118–126. Springer, Heidelberg (2002)
11. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases. University of California, Irvine, CA, Department of Information and Computer Science (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
12. Siciliano, R., Mola, F.: Multivariate Data Analysis through Classification and Regression Trees. In: Computational Statistics and Data Analysis, vol. 32, pp. 285–301. Elsevier Science, Amsterdam (2000)
13. Siciliano, R., Conversano, C.: Decision Tree Induction. In: Wang, J. (ed.) Encyclopedia of Data Warehousing and Data Mining, vol. 2, pp. 242–248. IDEA Group. Inc., Hershey, USA (2005)
14. Siciliano, R., Aria, M., Conversano, C.: Harvesting trees: methods, software and applications. In: Proceedings in Computational Statistics: 16th Symposium of IASC (COMPSTAT2004). Eletronical Edition (CD), Prague, August 23–27, 2004, Physica-Verlag, Heidelberg (2004)
15. Tutore, V.A., Siciliano, R., Aria, M.: Three Way Segmentation. In: Tutore, V.A., Siciliano, R., Aria, M. (eds.) Proceedings of Knowledge Extraction and Modelling (KNEMO06), IASC INTERFACE IFCS Workshop, September 4-6, 2006, Capri (2006)

Robust Tree-Based Incremental Imputation Method for Data Fusion

Antonio D'Ambrosio, Massimo Aria, and Roberta Siciliano

Dipartimento di Matematica e Statistica,
Università di Napoli Federico II, Italy
{antdambr,aria,roberta}@unina.it

Abstract. Data Fusion and Data Grafting are concerned with combining files and information coming from different sources. The problem is not to extract data from a single database, but to merge information collected from different sample surveys. The typical data fusion situation formed of two data samples, the former made up of a complete data matrix X relative to a first survey, and the latter Y which contains a certain number of missing variables. The aim is to complete the matrix Y beginning from the knowledge acquired from the X. Thus, the goal is the definition of the correlation structure which joins the two data matrices to be merged. In this paper, we provide an innovative methodology for Data Fusion based on an incremental imputation algorithm in tree-based models. In addition, we consider robust tree validation by boosting iterations. A relevant advantage of the proposed method is that it works for a mixed data structure including both numerical and categorical variables. As benchmarking methods we consider explicit methods such as standard trees and multiple regression as well as an implicit method based principal component analysis. A widely extended simulation study proves that the proposed method is more accurate than the other methods.

1 Introduction

Data Fusion and Data Grafting are concerned with combining files and information coming from different sources [12]. The problem is not to extract data from a single database, but to merge information collected from different sample surveys. The term *fusion* is used in this sense. The typical data fusion situation formed of two data samples, the former made up of a complete data matrix $\mathbf{X}$ relative to a first survey, and the latter $\mathbf{Y}$ which contains a certain number of missing variables. The aim is to complete the matrix $\mathbf{Y}$ beginning from the knowledge acquired from the $\mathbf{X}$. As a consequence, the Data Fusion can be considered as a particular case of data imputation framework, with the difference that in this case a group of instances is missing as they have not been observed. In literature, several approaches have been introduced dealing with Data Fusion:

– the **classical approach** such as regression models, general linear models and logistic regression [10];

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 174–183, 2007.

– the **implicit approach** based on the concept of similarity among the observations deriving from different sources [1];
– the **non parametric approach** that uses non standard regression techniques to impute missing values [4].

In the following, an innovative methodology for Data Fusion, based on a incremental imputation algorithm using tree-based models, is proposed.

The goal of this approach is the definition of the correlation structure which joins the two data matrices that are object of *the fusion*. A recursive use of robust segmentation trees validated by boosting iterations [7] will be considered. In this way, we define a tree-based incremental data fusion algorithm which proceeds, step by step, to the completion of the missing instances. This methodology allows to overcome the situation characterized by the presence of heterogeneous kinds of unobserved variables. As a matter of fact, the tree-based models give the possibility to work, at the same time, with both qualitative and quantitative data. The proposed algorithm has been implemented in MATLAB environment, as an additional module of *Tree Harvest Software* [3] [15]. Several analysis on simulated and real datasets show how this techniques can offer interesting results especially in comparison with the most famous classical and implicit approaches.

2 The Framework

Data Fusion problem may be formalized in term of two data files [1]. The first data file consists of a whole set of $p + q$ variables measured on n_0 subjects. This data file is called *donor file*. The second data file, usually named *receptor file*, consists in a subset of p variables measured on n_1 units (figure 1). So, the problem is to merge two different and independent databases.

We could imagine two independent surveys named survey A and survey B. Say that survey A has been collected in a particular supermarket, and say that survey

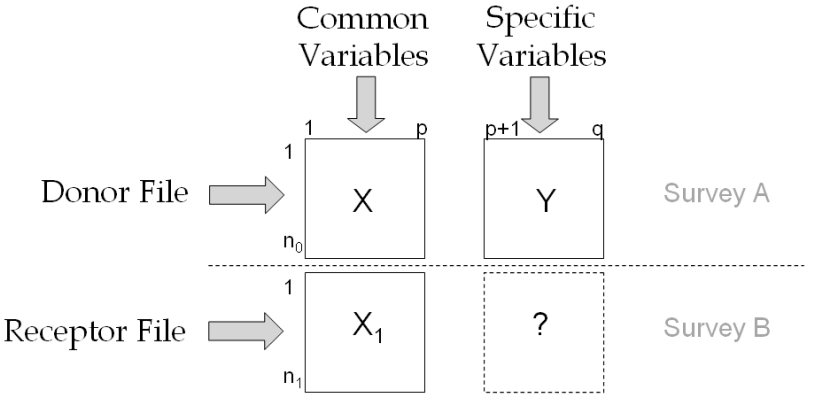


Fig. 1. Data Fusion mechanism

B has been collected in a different supermarket of the same chain located. The first set of variables X and X_1 are common to both supermarkets, whereas the other set Y is specific of Survey A. How would customer from second supermarket answers if we asked the same questions?

In the framework of Data Fusion we distinguish between explicit models and implicit models [12]. Explicit models are known as models based on variables, such as multiple regression, logistic regression, tree-based models, etc. With **explicit models**, a model is used to connect $\mathbf{Y}$ variables with the $\mathbf{X}$ variables in the donor file and then applying this model in the receptor file. **Implicit models** are models based on individuals (hot-deck imputation, k-nearest neighbours imputation, etc.), because for each individual belonging to the receptor file the k-nearest neighbours units of donor matrix are identified to transfer in some way the values of the $\mathbf{Y}$ variables to the receptor observation.

3 The Imputation Methodology

Our work follows the Incremental Non Parametric Imputation philosophy [13] [14]. Main idea of this explicit approach is to rearrange columns and rows of the data matrix according to a lexicographic ordering of the data (with respect to both rows and columns), that matches the order by value, corresponding to the number of missing data occurring in each record.

Main issue of the incremental approach is that following the positions defined by the index k, each column presenting missing values, at each turn, plays the role of dependent variable to be imputed by the complete set of variables with no missing values. Once that this variable is imputed it concurs to form the complete set of predictors used for the subsequent imputation.

Any missing column is handled using the tree-based model fitted to the current data which is iteratively updated by the imputed data. The imputation is incremental because, as it goes on, more and more information is added to the data matrix.

As a result, ensemble trees, by a boosting procedure, are used to impute data and the algorithm performs an incremental imputation of each single instance at time. Ensemble methods such as Boosting [7] allows to define a robust imputation procedure.

In the figure (2) the main steps of the imputation algorithm are described.

Let Y be a $N \times K$ matrix bearing missing data where y_k is the $k-$th variable of Y.

In the first step, the column k^* with the smallest number of missing data is found. This one plays the role of dependent variable. In the second step the columns with no missing values are sorted to obtain a complete matrix of p variables playing the role of predictors. In the following steps (i.e. the third and the fourth), the data are rearranged and a robust tree procedure is applied to estimate the missing values of the y_{k^*}. The algorithm iterates until all missing data are imputed.

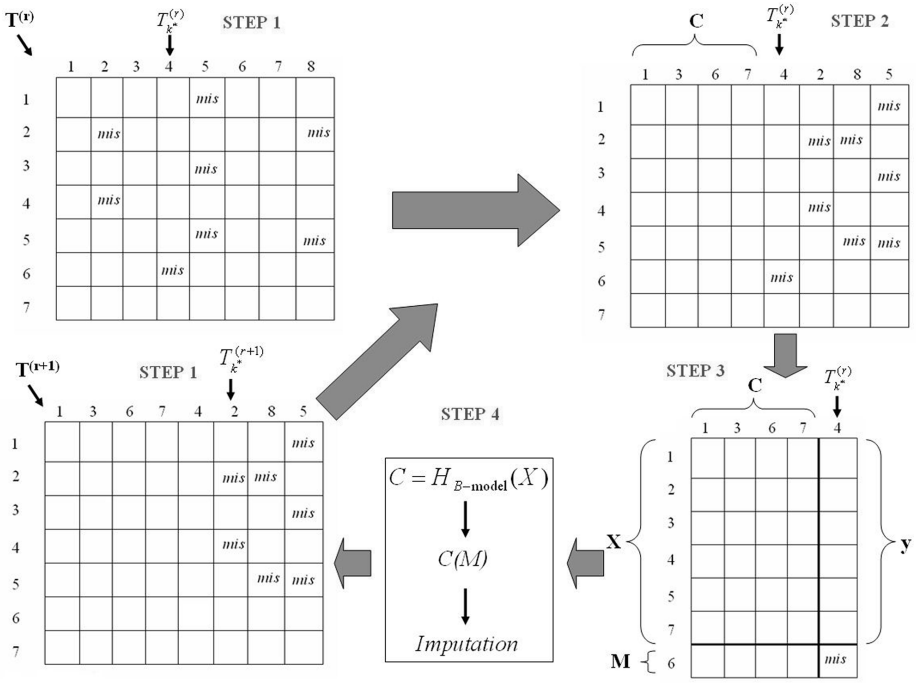


Fig. 2. Boosted Incremental Non Parametric Imputation Algorithm

4 Robust Incremental Imputation Algorithm for Data Fusion

The incremental approach, showed in the previous section, is integrated in a recursive methodology for data fusion that we call **Robust Incremental Imputation** (RTII).

Figure (3) describes the main steps of the proposed imputation algorithm for Data Fusion.

Let the donor file be formed by **A** and **B** blocks, let the receptor file be made by the **C** block, and let **D** be the block to impute.

Common variables $x_1, ..., x_p$ are represented by **A** and **C** blocks, whereas **B** block symbolizes specific variables $y_1, ..., y_q$.

4.1 The Main Step of RTII Algorithm

– **Step 0.**
  - Sort **B** block according to the dependence link with block **A**. Build a supervised tree for each y columns, then sort columns according to the previously obtained best tree.

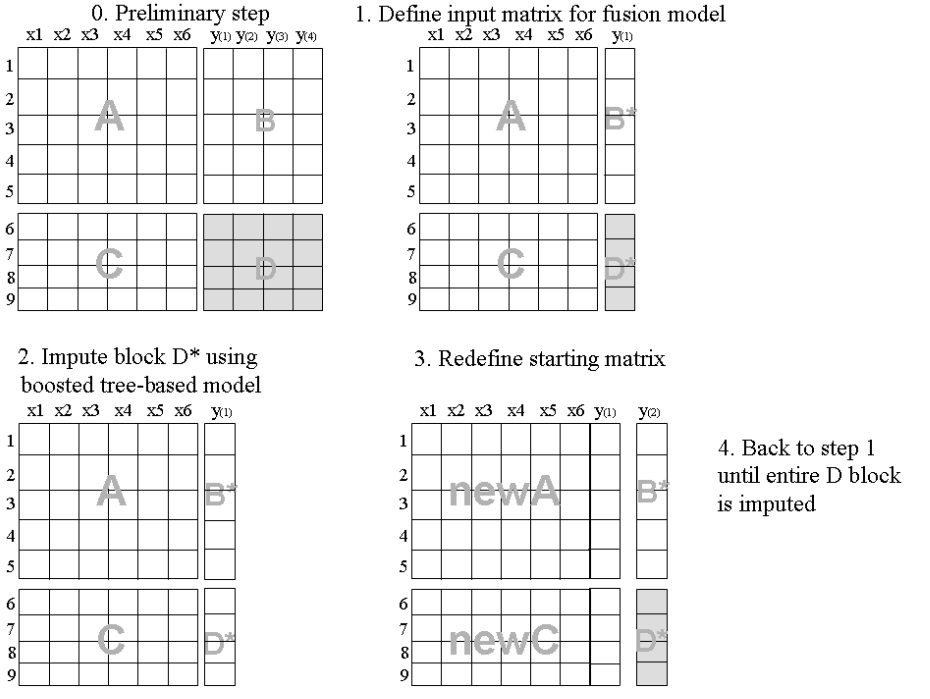


Fig. 3. Robust incremental imputation algorithm. (Grey blocks are missing values).

- **Step 1.**
  - Define the input matrix for fusion model; the first ordered y variable belonging to **B** block is the response and the complete variables (**A** block) are the predictors;
- **Step 2.**
  For $k = 1 : q$
  - Build a supervised tree with $V - fold$ AdaBoost iterations using blocks **A** as predictor and $\mathbf{B}^k$ as response variable, and use **C** to impute missing block $\mathbf{D}^k$;
  - Add $\mathbf{B}^k$ block to **A** to make the **newA** block and add the imputed block $\mathbf{D}^k$ to **C** one to form the **newB** block;
- **Step 3.**
  - The matrix is re-defined to impute a new column.
  - back to step 1 until all missing variables are completed;
- **Output.**
  - All variable belonging to **D** block are imputed

Preliminary step prepares the data matrix to the Fusion process. Specific variables are sorted according to their dependence link with common variables. A regression or classification tree is built for each specific variable, so these variables are sorted according to the best obtained root mean squared error (or the

lower misclassification ratio for categorical case). The first sorted variable belonging to the block of the specific ones play the role of response variable in the first iteration of the iterative imputation process. A supervised tree -as weak learner for V-fold AdaBoost iterations- is built, as a consequence the common variables belonging to the receptor file are used to impute the first missing variable. Both this variable and the specific one are included in the complete block and then the second sorted variable belonging to the specific ones is used as response variable. The iterative process ends when all variables are imputed.

5 Simulation Study

The performance of the proposed method based on *Robust Tree-based Incremental Imputation* (RTII) algorithm has been evaluated in a simulation study.

In this paper, we show an example regarding the imputation of numerical values, but RTII algorithm provides good performance also on categorical data (in particular, we use a boosted *stump* for binary case or a multiclass version of adaboost algorithm for multi-response case) and, in principal way, when we have a mixture of variables (both numerical and categorical) in the receptor file.

In this case, classical procedures cannot work, while RTII algorithm executes all iterations without problems since tree-based models deal with both numerical and categorical instances. The goal is to estimate punctual values of cases to be imputed, and then check the overall imputation procedure comparing the imputed variables with the control set in terms of their distribution. So, goodness imputation measures are the mean and standard deviation of imputed variables, the *root mean squared error* of used method, a measure of equality of mean between imputed and control variables using t-test, a measure of equality of variances between imputed and control variables using Fisher's test about equality of variances.

Performance of RTII algorithm has been compared with other methodologies such as Parametric Imputation via Multiple Regression (PI), Non Parametric Imputation by Standard Tree (Tree), PCA Fusion according to the Aluja et al.'s approach in [2].

Table (1) shows the simulation setting.

Simulation study have been defined thinking to reliable situations in which Data Fusion can be functional, i.e. when the donor file is a set of socio-economics variables (i.e., age, gender, income, job, etc.). For that reason, a simulated dataset was built using different random distributions for the set of common variables (Discrete uniform, Normal, Continue uniform), whereas specific variables were generated in both cases without relationship with common variables (according to normal and uniform distribution) and with linear link with other variables. Entire data set was randomly splitted in two sub-sets (donor file and receptor file), then the part of specific variables belonging to receptor file was deleted from the data set and used as control set to check the goodness of imputation.

Table 1. Description of the simulation setting

Simulation setting	
Donor file: 400 observations; Receptor file: 200 observations; Missing values: 1000	
Common variables	*Specific variables*
X_1 uniform in {18,65}	$Y_1 = k + 0.8X_2 - 0.2X_4$
X_2 uniform in [10,100]	$Y_2 = k + 0.2X_3 + 0.3X_1 + 3X_6$
X_3 uniform in [0,100]	$Y_3 = k + 0.6X_5 + 0.5X_2$
$X_4 \sim N(200, 30)$	Y_4 uniform in [50,250]
$X_5 \sim N(700, 80)$	$Y_5 \sim N(100, 10)$
X_6 *dummy* variable	

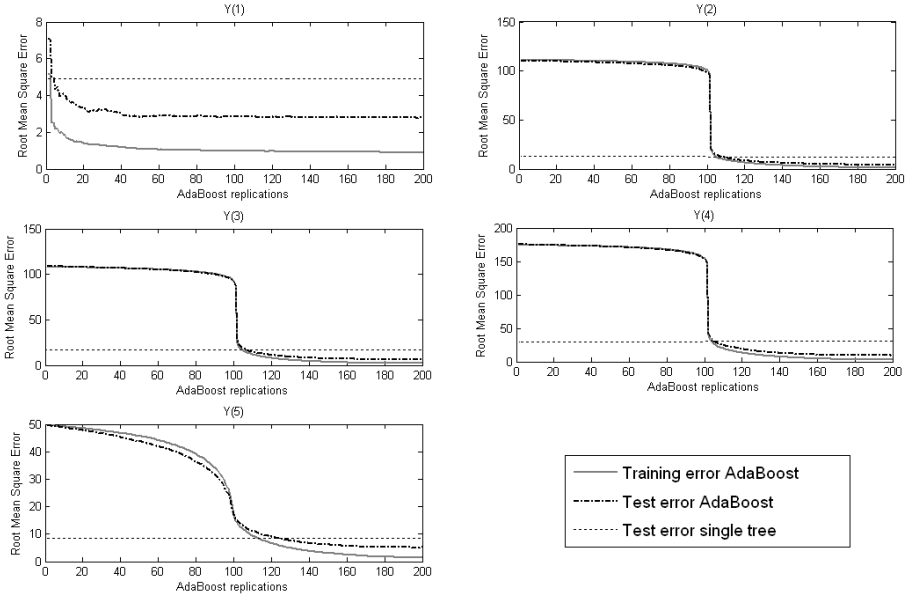


Fig. 4. Test error progress through AdaBoost iterations

Figure (4) shows test error progress through AdaBoost iterations. The error of boosted tree is always lower than error of single tree, and it seems to stabilize at the last tens of iterations, except for the first variable which reaches a stable rate of error after about 60 iterations.

Table (2) shows the results of our proposal in comparison with some standard Data Fusion techniques.

In bold the best results are underlined. They correspond to the mean and standard deviation values closer to the real ones. The performance of RTII algorithm is evaluated in terms of imputed mean, standard deviation, root mean squared error of the imputation, t-test for equality of means and F-test for

Table 2. Simulation study: main results

			Y_1	Y_2	Y_3	Y_4	Y_5
RTII	Mean		**105,550**	**206,470**	**313,250**	**145,700**	**100,580**
	Standard deviation		21,866	**54,624**	**69,789**	**52,529**	**39,739**
	Root Mean Squared Error		3,201	4,192	6,309	10,344	8,439
	Fisher's Variance Test	F stat	1,083	1,006	1,005	1,161	1,117
		P-value	0,287	0,482	0,486	0,146	0,218
	Compare Means Test	t stat	0,132	0,027	-0,084	-0,029	0,043
		P-value	0,447	0,489	0,467	0,488	0,483
Classical Tree	Mean		104,200	205,280	314,550	148,190	101,040
	Standard deviation		**22,034**	54,503	69,611	50,490	39,247
	Root Mean Squared Error		6,176	8,097	11,701	18,659	15,035
	Fisher's Variance Test	F stat	1,166	1,023	1,013	1,116	1,097
		P-value	0,140	0,436	0,463	0,220	0,258
	Compare Means Test	t stat	-0,446	-0,002	-0,073	0,385	0,375
		P-value	0,328	0,499	0,471	0,350	0,354
PCA (Aluja approach)	Mean		104,170	205,190	311,380	145,890	99,986
	Standard deviation		5,454	26,092	35,205	10,522	14,422
	Root Mean Squared Error		21,762	46,841	57,996	54,48	38,875
	Fisher's Variance Test	F stat	17,840	4,462	3,993	28,763	8,441
		P-value	0,000	0,000	0,000	0,000	0,000
	Compare Means Test	t stat	-0,673	-0,309	-0,392	0,027	-0,009
		P-value	0,251	0,379	0,347	0,489	0,497
Multiple regression	Mean		103,030	206,753	315,550	146,800	102,210
	Standard deviation		28,006	68,402	81,186	68,051	52,131
	Root Mean Squared Error		17,743	19,979	15,960	24,002	29,218
	Fisher's Variance Test	F stat	1,478	1,540	1,536	1,541	1,548
		P-value	0,003	0,001	0,001	0,001	0,001
	Compare Means Test	t stat	-0,118	0,001	-0,002	0,004	0,047
		P-value	0,453	0,500	0,499	0,498	0,481
True Mean			105,300	206,520	313,570	145,780	100,010
True Standard Deviation			23,035	55,114	70,349	56,432	41,901

equality of variances between imputed and control variables. These measures have been computed also for Parametric Imputation via Multiple Regression (PI), Non Parametric Imputation by Standard Tree (Tree), PCA Fusion according to Aluja et al.'s approach [2]. Always the root mean squared error of RTII algorithm is lower than other methods, and for all techniques the t-test is not significant.

As well-known in literature, a common problem concerning to classical imputation methods is to reduce, in a significant way, the variance of imputed distribution [10]. Considering this aspect, a method gives good results, when the imputation process doesn't involve a significant reduction of variance in

caparison with missing distribution. According to this point of view, Fishers test about equality of variances, has been used to compare the goodness of the different considered methodologies.

Reading the table (2) it can be noticed how all variables imputed via Multiple regression and PCA approach very often have a P-value of Fisher's Test close to zero, which drives to refuse the null hypothesis about equality of variances. On the contrary for the imputation via RTII algorithm and standard trees, the null hypothesis is never refused.

In our opinion, this is an important remark because tree-based methods, as non- parametric tools, are determinant not only for the imputation of missing values but above all in terms of variability reconstruction. Boosting algorithms make the estimate of missing variables more robust, as it can be seen looking at lower root mean squared errors.

6 Concluding Remarks

Data Fusion can be considered as a special case of data imputation where the values to be imputed are those allowing the merging between two different sample surveys [12]. This paper has provided a methodology for Data Fusion characterized by three features: first, it considers an explicit nonparametric method using a tree-based model; second, the recursive partitioning for data imputation makes use of an incremental approach where more and more information is added to the data matrix; third, the tree validation is robust since boosting iterations are performed. The overall method called *Robust Tree-based Incremental Imputation* presents two special advantages. First, it can be considered for a mixed data structure that includes both numerical and categorical variables. Second, it allows to reconstruct the imputed variable distribution in terms of both mean and variance. Indeed, in the simulation case study we have shown that this reconstruction is more accurate compared to other standard methods.

Acknowledgments. This work has been supported by MIUR funds 2005 and European FP6 Project iWebCare IST-4-028055. Authors are grateful to useful comments of anonymous referees.

References

1. Aluja-Banet, T., Morineau, A., Rius, R.: La greffe de fichiers et ses conditions d'application. Méthode et exemple. In: Brossier, G., Dussaix, A.M. (eds.) Enquêtes et sondages, Dunod, Paris, pp. 94–102 (1997)
2. Aluja-Banet, T., Rius, R., Nonell, R., Martínez-Abarca, M.J.: Data Fusion and File Grafting. Analyses Multidimensionelles Des Données. In: Morineau, A., Fernández Aguirre, K. (eds.) NGUS 97. 1 ed. París: CISIA-CERESTA, pp. 7–14 (1998)
3. Aria, M.: Un software fruibile ed interattivo per l'apprendimento statistico dei dati attraverso alberi esplorativi. Contributi metodologici ed applicativi. Phd thesis, University of Naples Federico II (2004)

4. Barcena, M.J., Tusell, F.: Enlace de encuestas: una propuesta metodológica y aplicación a la Encuesta de Presupuestos de Tempo. Qüestiio 23(2), 297–320 (1999)
5. Breiman, L., Friedman, J.H., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth (1984)
6. Eibl, G., Pfeiffer, K.P.: How to make AdaBoost.M1 work for weak base classifiers by changing only one line of the code. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS (LNAI), vol. 2430, Springer, Heidelberg (2002)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55(1) (1997)
8. Friedman, J.H., Popescu, B.E.: Predictive Learning via Rule Ensembles, Technical Report of Stanford University (2005)
9. Gey, S., Poggi, J.M.: Boosting and instability for regression trees. Computational Statistics and Data Analysis 50, 533–550 (2006)
10. Little, J.R.A., Rubin, D.B.: Statistical Analysis with Missing Data. John Wiley and Sons, New York (1987)
11. Petrakos, G., Conversano, C., Farmakis, G., Mola, F., Siciliano, R., Stavropoulos, P.: New ways to specify data edits. Journal of Royal Statistical Society, Series A, Part 2 167, 249–274 (2004)
12. Saporta, G.: Data fusion and data grafting. Computational Statistics and Data Analysis 38, 465–473 (2002)
13. Siciliano, R., Conversano, C.: Tree-based Classifiers for Conditional Missing Data Incremental Imputation. In: Proceedings of the International Conference on Data Clean, Jyvaskyla, May 29-31, 2002, University of Jyvaskyla (2002)
14. Siciliano, R., Aria, M., D'Ambrosio, A.: Boosted incremental tree-based imputation of missing data. Data Analysis, Classification and the Forward Search. In: Springer series in Studies in Classification, Data Analysis, and Knowledge Organization, Springer, Heidelberg (2006)
15. Siciliano, R., Aria, M., Conversano, C.: Harvesting trees: methods, software and applications. In: Proceedings in Computational Statistics: 16th Symposium of IASC. COMPSTAT2004, held Prague, August 23-27. 2004. Eletronical Edition (CD), Physica-Verlag, Heidelberg (2004)
16. van der Putten, P.: Data Fusion for Data Mining: a Problem Statement. In: Coil Seminar 2000, June 22-23, 2000, Chios, Greece (2000)
17. van der Putten, P., Kok, J.N., Gupta, A.: Data Fusion through Statistical Matching. MIT Sloan School of Management Working Paper No. 4342-02, Cambridge, MA (2002)

Making Time: Pseudo Time-Series for the Temporal Analysis of Cross Section Data

Emma Peeling and Allan Tucker

School of Information Systems Computing and Maths,
Brunel University, Uxbridge UB8 3PH, UK
{emma.peeling,allan.tucker}@brunel.ac.uk

Abstract. The progression of many biological and medical processes such as disease and development are inherently temporal in nature. However many datasets associated with such processes are from cross-section studies, meaning they provide a snapshot of a particular process across a population, but do not actually contain any temporal information. In this paper we address this by constructing temporal orderings of cross-section data samples using minimum spanning tree methods for weighted graphs. We call these reconstructed orderings *pseudo time-series* and incorporate them into temporal models such as dynamic Bayesian networks. Results from our preliminary study show that including pseudo temporal information improves classification performance. We conclude by outlining future directions for this research, including considering different methods for time-series construction and other temporal modelling approaches.

Keywords: Pseudo Time-Series, Cross-Section Data, Dynamic Bayesian networks, PQ Trees.

1 Introduction

Cross-section data record certain attributes across a population, thus providing a snapshot of a particular process but without any measurement of progression of the process over time. The progression of many biological and medical processes such as disease and development are inherently temporal in nature. However, many datasets associated with the study of such processes are often cross-section and the time dimension is often not incorporated or used in analyses. If we could reconstruct time-series from cross-section data, then it would allow many more datasets to be made available for building temporal models. For example, the vast amount of data based upon cross-section studies could be combined into *pseudo time-series* and temporal models built to explore the dynamics within.

In this paper, we update an existing algorithm that is based on minimum spanning tree methods for weighted graphs to reconstruct temporal orderings in cross-section data samples [8] and take this idea a step further by learning temporal models from such ordered data. We make use of Dynamic Bayesian Networks (DBNs) [4] to model the pseudo time-series. Bayesian networks model

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 184–194, 2007.

the data probabilistically and offer the advantage of being easily interpreted by non-statisticians. Previously, we have used DBNs extensively to model and classify data [12,13,15].

Converting cross-section studies into temporal data has the potential to allow many more novel analyses that were previously impossible, as well as allowing them to be incorporated into temporal models. For example, the forecasting of data based upon existing observations, and improved classification by using observations from 'earlier' points in the pseudo time-series. There is very little related work in this area and the closest studies involve combining a mixture of temporal and non-temporal data (such as cross-section study data) into single models. This work was mostly in the field of economics in the 1960s and 1970s [7] and more recently [11]. Another related research area involves the use of manifold techniques in conjunction with semi-supervised learning where only some cases are labelled with a class in a dataset. Unclassified datapoints are labelled based upon the distribution of other classified nearby points within the dataspace [1]. However, unlike our proposed method, this does not take into account the ordering of points in a sequence within the space.

This paper is organised as follows. Section 2.1 introduces pseudo time-series construction for cross-section data. Sections 2.2 and 2.3 describe Bayesian network models and how these can be applied to pseudo time-series of cross-sectional data. We apply our analysis to two cross-sectional datasets from the medical and biological domains, as described in section 3. Finally we present our conclusions in section 4.

2 Methods

2.1 Learning Pseudo Time-Series

In order to reconstruct time-series of multidimensional data samples we use an algorithm introduced by Magwene *et al.* [8] that is based on Minimum Spanning Tree (MST) methods for weighted graphs. Magwene *et al.* use their algorithm to estimate the temporal orderings of microarray gene expression data. The method uses a weighted graph where each vertex represents a data point. A pairwise distance function (such as Euclidean distance) is used to provide the edge weights. The approach is based upon the idea of modelling the dynamics of the data by visualising the trajectory of the data series through multidimensional space [10]. The weighted graph constructed from the data samples and its MST can be visualised in 2-D or 3-D space using principal coordinates methodology, thus providing a geometric interpretation of the data series. For example, Fig. 1 shows the weighted graph and its MST in 2-D for a small dataset consisting of 14 samples.

An initial ordering of the data samples can be constructed by converting the MST to a PQ tree. PQ trees [2] are a graph-structure device that can represent an ordering of points, and indicate which parts of the ordering are well-supported (Q-nodes) and which parts contain more uncertainty (P-nodes). Whilst the children of a P-node can be put into any order, children of a Q-node may be reversed

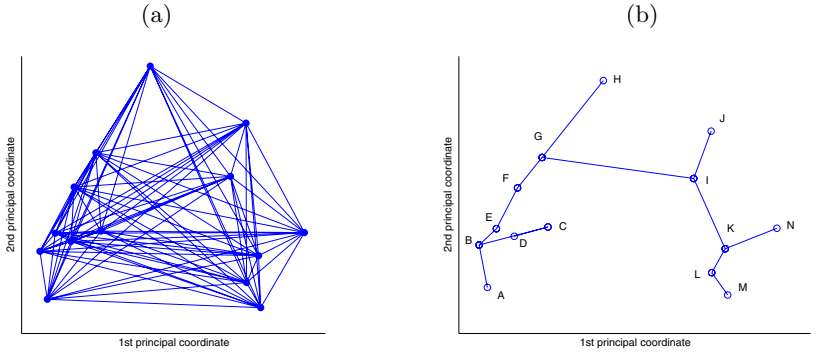


Fig. 1. From weighted graph to minimum spanning tree: (a) the weighted graph for a small dataset consisting of 14 sample points. Each edge is weighted with the distance between its corresponding vertices. Weights are not shown on this graph. (b) the MST for the weighted graph, with vertices (data points) labelled A-N.

in order but may not otherwise be reordered. The diameter path of the MST is used as the main Q-node of the PQ tree - the backbone of the reconstructed ordering. Branches off the diameter path are added as P and Q nodes to the main Q node. Therefore, the constructed PQ-tree represents a partial ordering of the data samples. The PQ-tree for the weighted graph and MST in Fig. 1 is shown in Fig. 2a.

In practice, we have found that many PQ-Tree orderings contain a high degree of uncertainty in the P-nodes and on larger datasets an exhaustive search is not

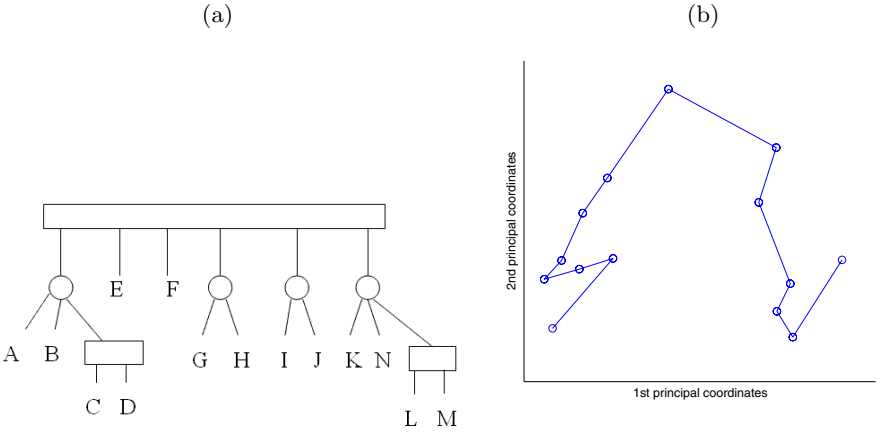


Fig. 2. (a) Shows a PQ-tree for a small dataset consisting of 14 points, labelled A-N. The weighted graph and MST for this dataset is shown in Fig. 1 (b) Final ordering shown as a trajectory in 2-D principal coordinates after hill-climb search on PQ-tree.

feasible. To resolve this we use a hill-climbing search to optimise the ordering at each P-node by minimising the total length of the sequence based upon the Euclidean distance. This process produces a full ordering of the samples upon which we can base further analysis of the data using temporal models (see Fig. 2b). We refer to this ordering as the *pseudo* time-series or temporal ordering.

2.2 Bayesian Network Models

Here we use Dynamic Bayesian Networks (DBNs) [4] to model the temporal relationships in pseudo time-series though in future work we intend to explore various time-series and sequence models. Bayesian Networks (BNs) [6] are probabilistic models that can be used to model data transparently. This means that it is relatively easy to explain to non-statisticians how the data are being modelled unlike other 'black box' methods. DBNs are an extension of BNs to handle the sort of relationships that are found in time-series.

A BN consists of two components. The first is a Directed Acyclic Graph (DAG) consisting of links between nodes that represent variables in the domain. The second component is a set of conditional probability distributions associated with each node. These probability distributions may be modelled using discrete (tables) or continuous (e.g. Gaussian) distributions. If there is a link from node A to another node B, then A is said to be a *parent* of B, and B is a *child* or *descendant* of A. The directed arcs between nodes indicate the existence of conditional independence relations between variables, whilst the strengths of these influences are quantified by the conditional probabilities. It should be noted that for any one network there will be a family of other networks that encode the same independence relations. These are known as the *equivalence class*. In DBNs, BNs are extended with additional nodes that are used to represent the

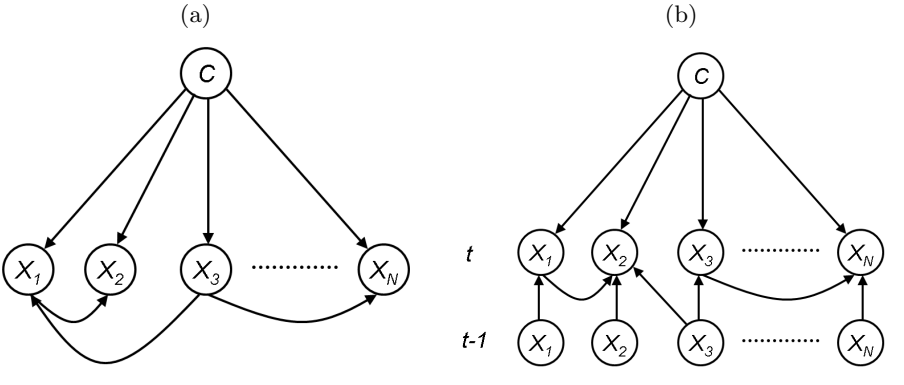


Fig. 3. (a) A typical BNC with n variables. The class node C represents a classification of each data sample. (b) In a DBNC model, links occur between variables over time and within the same time slice. This DBNC has n variables and 2 time slices, t and $t-1$.

variables at differing time slices. Therefore links occur between nodes over time and within the same time slice. BNs and DBNs can be used to classify data by simply adding a class node that represents the classification of a data sample, as shown in Fig. 3. These classifiers are known as Bayesian Network Classifiers (BNCs) [3]and Dynamic Bayesian Network Classifiers (DBNCs) [14].

2.3 Pseudo Temporal Model Construction

Using the pseudo temporal ordering of the data samples we can build a DBNC model of the dataset. The idea is that the inclusion of temporal information should improve model performance. In the remainder of the paper we investigate this claim using both a medical and a biological dataset concerning disease development and compare the classification performance of a BNC and a DBNC based on the pseudo time-series. The whole process for creating a pseudo temporal model of cross-section data is described in Algorithm 1.

Algorithm 1. Pseudo temporal model construction

1: Input: Cross-section data
2: Construct weighted graph and MST
3: Construct PQ tree from MST
4: Derive pseudo time-series from PQ-tree using a hill-climb search on P-nodes to minimise sequence length
5: Build DBNC model using pseudo temporal ordering of samples
6: Output: Temporal model of cross-section data

3 Experiments and Results

In this section we present results from experiments to investigate whether the inclusion of pseudo temporal information improves model performance. We consider two cross-section datasets: B-cell lymphoma microarray data and glaucoma visual field data and focus on the classification of healthy and diseased samples.

For both datasets we construct pseudo temporal orderings of the disease progression using Euclidean distance weighted graphs, PQ-trees and hill-climbing search to optimise sequence length (as described in section 2.1). We then compare the classification performance of a DBNC learnt from the pseudo time-series against a BNC learnt with no temporal information, to see if the temporal relationships improve classification. For the B-cell dataset experts were able to provide us with an assumed ordering of the data samples according to disease progression so we are able to compare this with the pseudo ordering.

In order to assess the performance of a classifier and to compare different classifiers, it is common practice to use Receiver Operator Characteristic (ROC) curves [5]. An ROC curve allows one to view graphically the performance of a classifier by plotting the sensitivity (which in our case is the proportion of correctly classified positives) against the specificity (the proportion of misclassified

positives) for varying thresholds used by the Bayesian classifiers. The perfect classifier would have a ROC curve that follows the top-left corner of the unit square, whereas the worst situation would be a classifier whose curve follows the diagonal. Real applications will usually show curves between these two extremes. A global measure of the classifier performance, often used in classification problems, is the Area Under the ROC Curve (AUC). This will be some value between 0.5, associated to the diagonal of the square, and 1, corresponding to the curve that follows the top-left corner. We use this to score the classifiers learnt from the two real-world datasets.

A leave-one-out cross-validation approach was adopted for the B-cell data due to the small number of samples for this dataset, whereas for the visual-field data, we have two independent datasets. A cross section dataset which is used to train pseudo time-series models from and a set of short-time series from a longitudinal study is used to test the models. We repeat the model learning and classification for each fold 10 times using a bootstrap approach in order to get confidence intervals on the mean AUC. For the B-Cell data which has 3 classes, the mean of the AUCs for all comparisons is calculated. In other words, the mean AUC for classifying class 0 from 1 or 2, class 1 from 0 or 2, and class 2 from 0 or 1 is used. See [9] for how ROC analysis can be performed on multi-class problems.

Sections 3.1 and 3.2 present our results for the B-cell lymphoma and glaucoma datasets respectively.

3.1 B-Cell Lymphoma

Dataset. This dataset concerns a short cross-section study measuring gene expression for 26 patients across a set of B-cell lymphomas and leukaemias. For each patient we know whether the disease is present and how advanced it is; each sample is classified into one of three classes - healthy, diseased (early) or diseased (late).

Learning the Pseudo Time-Series. Using the expert (biologists) ordering, we labelled the samples from 1 to 26. Table 1 lists the ordering given by biologists, the initial PQ-tree and the PQ-tree obtained after hill-climb search on its P-nodes. We can see that the biologists ordering is optimal in terms of sequence length. The hill-climb improves on the initial PQ-tree both in terms of sequence length and similarity to the biologists ordering. Figure 4a shows a plot of the data samples, labelled by classification in 2-D principal coordinates. Figure 4b shows the multidimensional trajectory 3-D plot of the ordering produced by the PQ-tree/hill-climb. We can see that the trajectory has a clear start and end point and that the classes seem to be dispersed over this trajectory. The shape of this trajectory along with the similarity between the biologists ordering and the PQ-Tree/hill climb ordering imply that the natural ordering can indeed be learnt from such data.

Temporal Models. Next we applied a number of classifier models to the dataset and compared the results. We considered a BNC (no temporal information), and DBNCs for the expert ordering and the pseudo ordering. The results

Table 1. B-cell data sample orderings obtained from biologists, PQ-tree and PQ-tree with hill-climb search on P-nodes

	Ordering	Sequence length
Biologist	1-26	512.0506
PQ-tree	1-6,7,9,8,11,10,12-18,26,19,21,20,22-25	528.9907
PQ-tree and hill-climb	1-18,26,19-25	521.1865

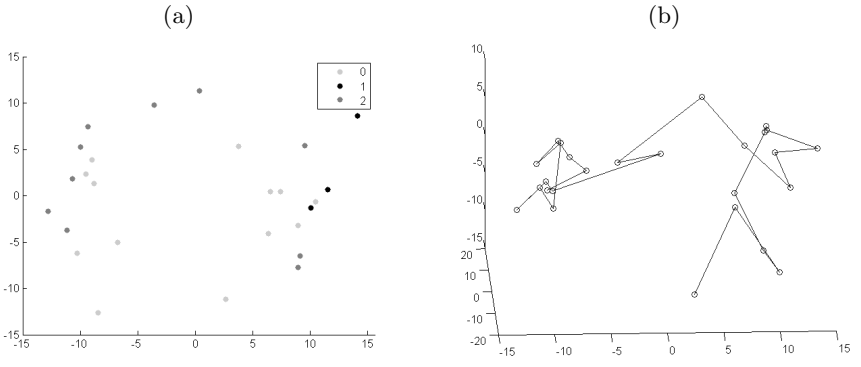


Fig. 4. (a) Shows the B-cell data samples plotted in 2-D principal coordinates, with their classification. Class 0 is healthy, Class 1 is diseased (early) and Class 2 is diseased (late). (b) shows the B-cell temporal ordering as a 3D trajectory. Notice that there is a clear start and end point.

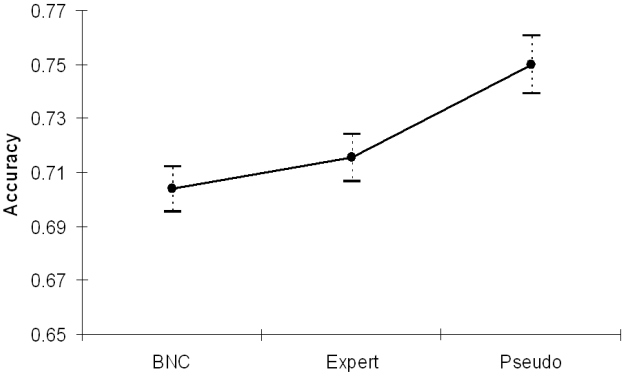


Fig. 5. Model comparison for B-cell data: Mean Area Under the ROC Curve: BNC, Pseudo DBNC

of the ROC analysis of these classifiers show that the DBNC architecture is best for this dataset, as the pseudo and biologists ordering gain the highest mean AUC (see Figure 5). Notice that the expert ordering gives a slightly improved AUC but that the pseudo time-series results in a significantly improved AUC. This graph implies that if we can find a good ordering on cross-section data then we can indeed learn useful temporal relationships to improve non-temporal models. We also generated a random ordering and learnt a DBNC architecture to see to what extent spurious relationships could affect the classification. Indeed a random ordering DBNC decreases the performance to lower than that for a BNC (an accuracy of 0.7 and AUC of 0.84), so it is essential that an accurate temporal ordering is used to avoid spurious relationships being discovered.

3.2 Glaucoma and Visual Field Deterioration

Data. Our second application concerns Visual Field (VF) data. We have a large cross-section study on 162 people concerning glaucoma, an eye disease that leads to the progressive deterioration of the field of vision. For each patient the dataset contains a visual field test, which assesses the sensitivity of the retina to light. The data for each patient is classified into one of two classes: healthy or glaucomatous based upon clinical observation of the tests. We use this dataset as the training set to learn the pseudo time-series from.

For testing the pseudo time-series, we use another VF dataset. This contains a set of time-series for 24 patients attending the Ocular Hypertension Clinic at Moorfields Eye Hospital. All of these patients initially had normal VFs but developed reproducible glaucomatous VF damage in a reliable VF in their right eye during the course of follow-up. This dataset can be used as an independent and unseen data set to evaluate the learnt pseudo temporal models.

Learning the Pseudo Time-Series. Using an arbitrary ordering (as we have no known expert ordering for this data), we generate the PQ-Tree and calculate the distance of the PQ-tree obtained after hill-climb search on its P-nodes. This was found to be 4462.6 as opposed to 5244.3 prior to the hill-climb. Figure 6 shows the allocation of classes in the first two principal coordinates.Notice that there is an obvious starting point in the dataspace where the healthy patients are clustered and that these 'fan' out towards different regions in the space for the glaucomatous patients.

Temporal Models. Next we applied a BNC and a DBNC using the pseudo time-series to the dataset and compared the results. Figure 7 shows the mean AUC and the confidence limits for the BNC and the Pseudo DBNC. It is clear that the mean AUC improves and the variation reduces when the pseudo temporal links are included. This implies that, whilst there is increased risk of spurious relationships when looking for pseudo temporal links, if a good ordering can be found the additional temporal links can improve accuracy.

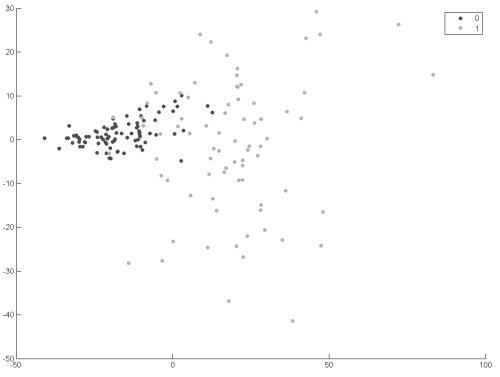


Fig. 6. VF data in 2D space showing the classification of glaucoma

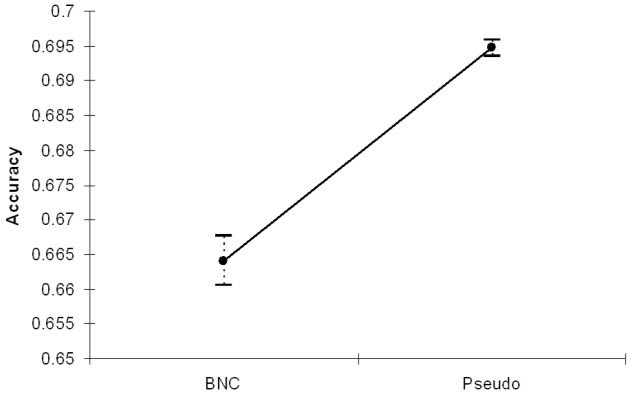


Fig. 7. VF data Area Under Curve Comparison: BNC, Pseudo

4 Conclusions and Future Work

In this paper we propose a method for incorporating temporal information into cross-section data models. Using an updated existing algorithm based on minimum spanning tree methods for weighted graphs we construct pseudo time-series and include them in temporal models such as dynamic Bayesian network classifiers. In this study our results on medical and biological datasets concerning disease development show that including pseudo temporal information does improve classification performance. However, this research is very preliminary - whilst we have evaluated the methodology on two datasets, they are of relatively small size and the pseudo-temporal model accuracies improve on the static models by only a few percentage points.

There are many directions for future research. The results show that finding an accurate pseudo time-series is crucial - an incorrect ordering of samples

can worsen the performance of the classifier and increase the risk of spurious correlation. Different methods for ordering the data such as fitting polynomials through the Euclidean space will be considered in future work. Additionally, we only considered Bayesian networks to model the data. In future, we will look at different temporal models to see which best capture the dynamics. For example, a better way to model the pseudo time-series could be using continuous time models. There are also many issues concerning bias in different studies and experiments. In other words, how is it possible to combine several studies that sit in very different areas of Euclidean space due to experimental bias. Lastly, it would be interesting to see if we could model several trajectories that cover different areas of the dataspace. For example, the VF data looks as though there are different regions of the dataspace that patients move into, based upon different forms of glaucoma. Rather than fitting one trajectory based upon a single ordering, it may be better to explore fitting numerous trajectories.

Acknowledgements

We would like to thank Paul Kellam for supplying the B-Cell data and David Garway-Heath and Nick Strouthidis for supplying the visual field datasets.

References

1. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning for examples. Technical Report, Univ. of Chicago, Department of Computer Science (TR-2004-06) (2004)
2. Booth, K.S., Lueker, G.S.: Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. Journal of Computer and System Sciences (13), 335–379 (1976)
3. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29, 131–163 (1997)
4. Friedman, N., Murphy, K.P., Russell, S.J.: Learning the structure of dynamic probabilistic networks. In: Proceedings of the 14th Annual Conference on Uncertainty in AI, pp. 139–147 (1998)
5. Hand, D.J.: Construction and Assessment of Classification Rules. Wiley, Chichester (1997)
6. Heckerman, D., Geiger, D., Chickering, D.: Learning bayesian networks: The combination of knowledge and statistical data. In: KDD Workshop, pp. 85–96 (1994)
7. Hoch, I.: Estimation of production function parameters combining time-series and cross-section data. Econometrica 30(1), 34–53 (1962)
8. Magwene, P.M., Lizardi, P., Kim, J.: Reconstructing the temporal ordering of biological samples using microarray data. Bioinformatics 19(7), 842–850 (2003)
9. Patela, A.C., Markey, M.K.: Comparison of three-class classification performance metrics: a case study in breast cancer cad. 5749 (2005)
10. Rifkin, S.A., Kim., J.: Geometry of gene expression dynamics. Bioinformatics 18(9), 1176–1183 (2002)
11. Strauss, W.J., Carroll, R.J., Bortnick, S.M., Menkedick, J.R., Schultz, B.D.: Combining datasets to predict the effects of regulation of environmental lead exposure in housing stock. Biometrics 57(1), 203–210 (2001)

12. Tucker, A., Garway-Heath, D., Liu, X.: Bayesian classification and forecasting of visual field deterioration. In: The Proceedings of the Ninth Workshop on Intelligent Data Analysis in Medicine and Pharmacology and Knowledge-Based Information Management in Anaesthesia and Intensive Care (2003)
13. Tucker, A., Liu, X.: Learning dynamic bayesian networks from multivariate time series with changing dependencies. Intelligent Data Analysis– An International Journal 8(5), 469–480 (2004)
14. Tucker, A., 't Hoen, P.A.C., Vinciotti, V., Liu, X.: Bayesian network classifiers for time-series microarray data. Lecture Notes in Computer Science (3646), 475–485 (2005)
15. Tucker, A., Vinciotti, V., Garway-Heath, D., Liu, X.: A spatio-temporal bayesian network classifier for understanding visual field deterioration. Artificial Intelligence in Medicine (34), 163–177 (2005)

Recurrent Predictive Models for Sequence Segmentation

Saara Hyvönen, Aristides Gionis, and Heikki Mannila

Helsinki Institute for Information Technology, Department of Computer Science, University of Helsinki, Finland

Abstract. Many sequential data sets have a segmental structure, and similar types of segments occur repeatedly. We consider sequences where the underlying phenomenon of interest is governed by a small set of models that change over time. Potential examples of such data are environmental, genomic, and economic sequences. Given a target sequence and a (possibly multivariate) sequence of observation values, we consider the problem of finding a small collection of models that can be used to explain the target phenomenon in a piecewise fashion using the observation values. We assume the same model will be used for multiple segments. We give an algorithm for this task based on first segmenting the sequence using dynamic programming, and then using k-median or facility location techniques to find the optimal set of models. We report on some experimental results.

1 Introduction

Sequential data occur in many applications and finding out the structure of the underlying process that generates such data is one of the key tasks in data mining and data analysis in general. In practice many prediction or modeling tasks can have a segmental structure: different models are valid in different segments. For example, high solar radiation implies clear skies, which in the summer means warm temperatures and in the winter cold ones. As another example, the inheritance mechanism of recombinations in chromosomes mean that a genome sequence can be explained by using a small number of ancestral models in a segment-wise fashion. In these examples, the model used to explain the target variable changes relatively seldom, and has a strong effect on the sequence. Moreover, same models are used repeatedly in different segments: the summer model works in any summer segment, and the same ancestor contributes different segments of the genome.

To find such structure, one must be able to do segmentation based not on the target to be predicted itself, but on which model can be used to predict the target variable. In this paper we describe an approach that can be used to search for structure of this type. Given a model class $\mathcal{M}$, we search for a small set of h models from $\mathcal{M}$ and a segmentation of the sequence into k segments such that the behavior of each segment is explained well by a single model. We assume that $h \ll k$, i.e., the same model will be used for multiple segments. Previously, the

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 195–206, 2007.

idea for searching for recurrent models has been used in the context of finding piecewise constant approximations [6]. Here we use the approach to arbitrary predictive models; this requires considerably different techniques. For any but the simplest model class the problem of finding the best h models is an NP-hard task, so we have to resort to approximate techniques.

Given a sequence and a class of models, we first use dynamic programming to find a good segmentation of the sequence into k segments. Thus each segment will have its unique predictive model. After that, from the k models found in the segmentation step we select a smaller number of h models that can be used to model well the whole sequence. Selecting a smaller number of models is done using the k-median [19] or the facility location approach [12], depending on whether we wish to fix the number of models in advance or allow the method to decide a good number of models. The method for finding the model describing a single segment depends, of course, on the model class $\mathcal{M}$.

We have applied the method to two sets of real data, meteorological measurements, and haplotypes in the human genome. The experimental results show that the method produces intuitive results and is reasonably efficient.

The rest of this paper is organized as follows. We give a brief overview of related work in Section 2. In Section 3 we present the notation and we give the problem definition. The basic algorithms for both segmentation and model selection are described in Section 4. Empirical results on two different data sets are discussed in Section 5, and a short conclusion is presented in Section 6.

2 Related Work

Segmentation of sequences is a widely studied problem, starting from the classical paper of Bellman [1]. Segmentation has been applied to various applications, including time series [15], stream data from mobile devices [10] and genomic data [17,18] to mention a few.

Recently work has been done on finding segments with different complexities [8] and segmenting multivariate sequences in such a way that the data in each segment has a low dimensional representation [2]. Also a lot of work has been done in finding more efficient algorithms [7,22,21].

Our work is related to the (k, h)-segmentation problem [6]: given a sequence of length n, find a good way to divide it into k segments, each of which comes from h different sources. For example, one may wish to segment a time series into k segments in such a way that in each segment the time series can be approximated well by one of the h constants. Our work extends this approach: we not only wish to model the sequence in a piecewise manner, but predict the target sequence in a piecewise fashion using the observation values and one of h models from some model class $\mathcal{M}$.

Segmentation based on sequence modeling has been done using specific classes of models, such as burstiness [16] or Hidden Markov models [20]. One should note that our approach differs from these in that while e.g., HMMs associate segments with hidden variables and corresponding emission probabilities, we use different

kinds of models, e.g., regression models, predictive models, etc. Moreover, unlike HMMs we do not have transition probabilities between models.

3 Problem Definition

Data. We are given a sequence of n data points $\mathbf{D} = (\mathbf{X}, \mathbf{y})$. The i-th data point, $i = 1 \ldots n$, consists of a vector of d *observation values* $\mathbf{X}[i]$ and an *outcome value* $\mathbf{y}[i]$. In other words, $\mathbf{X}[i]$, $i = 1 \ldots n$ is a d-dimensional vector while $\mathbf{y}[i]$ is a scalar. Depending on the application $\mathbf{X}[i]$ and $\mathbf{y}[i]$ may both take arbitrary real values, or both be binary, or it is possible that $\mathbf{X}[i]$ is real valued while the outcome values are binary, i.e., $\mathbf{y}[i] \in \{0, 1\}$.

Segmentations. For $1 \leq i \leq j \leq n$ we write $\mathbf{D}[i..j] = (\mathbf{X}[i..j], \mathbf{y}[i..j])$ to denote the subsequence of the input sequence between the i-th and j-th data points (so $\mathbf{D}[1..n] = \mathbf{D}$). We consider segmentations of the input sequence. A k-segmentation is a partition of the input sequence into k continuous and non-overlapping segments. More formally, a k-segmentation of $\mathbf{D}$, denoted by $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_k)$, is defined by $k+1$ *boundary points* $1 = b_1 \leq b_2 \leq \ldots \leq b_k \leq b_{k+1} = n + 1$ such that the j-th segment $\mathbf{D}_j$ is the subsequence $\mathbf{D}[b_j..b_{j+1} - 1]$.

Prediction. A *model* M is a function $M : \mathbb{R}^d \to \mathbb{R}$. A *model class* $\mathcal{M}$ is a set of models. The set of models $\mathcal{M}$ may be parameterized by a set of parameters θ (e.g., regression models), or it may consists of a finite set of fixed models.

We want to find models in the model class $\mathcal{M}$ that correctly predict the outcome value $\mathbf{y}[i]$ of the i-th data point, given the observation vector $\mathbf{X}[i]$. Given a subsequence $\mathbf{D}[i..j]$ and a model M we define the *prediction error* of model M on $\mathbf{D}[i..j]$ as

$$E(\mathbf{D}[i..j], M) = \sum_{t=i}^{j} (M(\mathbf{X}[t]) - \mathbf{y}[t])^2. \tag{1}$$

The task of predicting a binary outcome value ($\mathbf{y}[i] \in \{0, 1\}$) is called *binary classification*. If $\mathbf{y}[i]$ is binary and $M(\mathbf{X}[i])$ outputs binary values, the least squares error defined by equation 1 is simply the number of misclassifications.

For many commonly used model classes $\mathcal{M}$ we can compute in *polynomial time* the model $M^* \in \mathcal{M}$ that minimizes the error in equation (1). For example for the class of linear models the optimal model can be computed using least squares. For probabilistic models we can compute the maximum likelihood model. Even for cases that finding the optimal model is computationally difficult (e.g., decision trees [13]) effective heuristics are usually available. So in this paper we assume that we can always find a good model for a given subsequence.

We now define our first problem: segmenting a sequence and finding per-segment models in order to minimize the total prediction error.

Problem 1. Given an input sequence $\mathbf{D}$, a model class $\mathcal{M}$, and a number k, segment $\mathbf{D}$ into k segments $(\mathbf{D}_1, \ldots, \mathbf{D}_k)$ and find corresponding models $\{M_1, \ldots, M_k\} \in \mathcal{M}$ so that the overall prediction error $\sum_{j=1}^{k} E(\mathbf{D}_j, M_j)$ is minimized.

Recurrent models. In Problem 1 we search for a k-segmentation of the input sequence and we allow each segment to be fitted with a separate model. A more demanding task is to search for *recurrent predictive models*. In this case, we want to obtain a k-segmentation, as well, but we allow only a small number of h distinct models ($h < k$). Thus, some of the models have to be used to fit more than one segments. More formally we define the following problem.

Problem 2. Consider a sequence $\mathbf{D}$, a model class $\mathcal{M}$, and numbers k and h. We want to find a k-segmentation of $\mathbf{D}$ into k-segments $(\mathbf{D}_1, \ldots, \mathbf{D}_k)$, h models $M_1, \ldots, M_h \in \mathcal{M}$, and an assignment of each segment j to a model $M_{m(j)}$, $m(j) \in \{1, \ldots, h\}$ so that the prediction error $\sum_{j=1}^{k} E(\mathbf{D}_j, M_{m(j)})$ is minimized.

Note that if the model class $\mathcal{M}$ is large, there is a serious risk of overfitting. Therefore, when applicable, the usage of cross-validation is recommended.

4 Algorithms

We first discuss the components of our algorithm and then in Section 4.4 we describe how we put all the components together.

4.1 Dynamic Programming

Problem 1 can be solved optimally in polynomial time by dynamic programming [1]. The main idea is to perform the computation in an incremental fashion using an $(n \times k)$-size table A, where the entry $A[i, p]$ denotes the error of segmenting the sequence $\mathbf{D}[1..i]$ using at most p segments. The computation of the entries of table A is based on the equation

$$A[i, p] = \min_{1 \le j \le i} \left(A[j-1, p-1] + E(\mathbf{D}[j..i], M^*_{ji}) \right), \tag{2}$$

where $E(\mathbf{D}[j..i], M^*_{ji})$ is the minimum error that can be obtained for the subsequence $\mathbf{D}[j..i]$, that is, M^*_{ji} is the optimal model for that subsequence.

As we have already mentioned, the optimal model M^*_{ji} and thus also the error $E(\mathbf{D}[j..i], M^*_{ji})$ can be computed in polynomial time. If $T(|i-j|)$ is the time required to compute M^*_{ji}, then the overall running time of the dynamic programming algorithm is $O(n^2(k + T(n)))$. In some cases the computation can be speeded up using precomputation and the fact that we compute many models simultaneously. For instance, for computing least-square models, even though computing a single model M^*_{ji} requires time $O(|i-j|)$, computing all $\frac{n(n-1)}{2}$ models requires time $O(n^2)$ (i.e., constant amortized time) leading to overall running time for the dynamic programming equal to $O(n^2 k)$.

4.2 Clustering Algorithms

Let us turn to Problem 2. Recall that we want to find a k-segmentation of the sequence, h models ($h < k$), and an assignment for each segment to one of those

h models. We start by performing a k-segmentation $(\mathbf{D}_1, \ldots, \mathbf{D}_k)$ with k models $\{M_1, \ldots, M_k\}$, as in described in Section 4.1. We then *fix* the segmentation and proceed with selecting the best h models and finding the assignment from segments to models. This approach of first segmenting and then clustering is motivated by previous work [6], in which it has been shown to be a good approach. We consider the following alternatives.

k-median algorithm. We select the h models to be among the set $\{M_1, \ldots, M_k\}$. For each segment $\mathbf{D}_i$, $i = 1, \ldots, k$ and each model M_j, $j = 1, \ldots, k$, we first compute the error $c_{ij} = E(\mathbf{D}_i, M_j)$. The problem of selecting the best h models is now transformed to the k-median problem, which is defined as follows: Given n objects (in our case the k segments), m service points (in our case the k models), and a cost function c_{ij} for each object-service point pair, select h service points so that the total cost of each object to its nearest selected service point is minimized. The k-median problem is NP-hard [5]. If the costs c_{ij} satisfy the triangle inequality, then it can be approximated within a constant factor [3]. In our case however the cost between two service points (models) or two objects (segments) is not defined. In the general case, where the costs do not form a metric the situation is more complex: the best known approximation algorithm is the one provided by Lin and Vitter [19], which provides an $(1+\frac{1}{\epsilon})$-approximate solution but it uses $O(k(1+\epsilon)\log n)$ medians instead of k.

A simple scheme for solving the k-median problem is provided by a k-means type iterative algorithm: start by picking k service points at random to be the set of medians, then alternatively assign each object to the closest median and recompute the median. The new median is computed by taking all objects assigned to the old median and finding the service point for which the total cost of these objects to the chosen service point is minimized. Though this scheme only converges to a local minimum, in practice it often finds a good solution efficiently. Notice that for small values of k and h, the problem can be solved optimally by exhaustive search — considering all $\binom{k}{h}$ model combinations.

Facility location algorithm. The facility location problem differs from the k-median on the fact that there is no restriction on the number of service points (facilities) to be selected. Instead, each possible service point has an "opening" cost, and the objective is to minimize the total cost of opening service points plus the sum of costs from each point to its closest opened service point. More formally, we are given a set of service points F, a set of objects C, costs f_j for opening a service point $j \in F$, and distances d_{ji} for each $j \in F$ and $i \in C$. The task is to find a set of service points to open, that is, find $S \subseteq F$ to minimize $Z(S)$, where $Z(S) = \sum_{i \in C} \min_{j \in S} d_{ji} + \sum_{j \in S} f_j$.We use the facility location problem to give a minimum description length (MDL) interpretation to the model selection process. In particular, the set of service points is the set of models $\{M_1, \ldots, M_k\}$ and the set of objects are the segments $\mathbf{D}_i$, $i = 1, \ldots, k$ found in the k-segmentation phase. The distance d_{ji} from j to i is set to be the error $E(\mathbf{D}_i, M_j)$, and the cost f_j of using a additional model j (opening a service point) is set to be the description length of the model j, measured in

bits. Consequently, the task is to find the optimal set of models that minimizes the overall description of the selected models plus the description of the data given the models. Since models in a model class are usually of equal complexity, it is meaningful to have a constant cost for all service points. The choice of the cost depends on the application. Higher cost means a larger penalty for adding a new model.

In the general case, when the costs d_{ji} do not satisfy the triangle inequality, the facility location problem can be approximated to $O(\log n)$ factor [12], and no better approximation factor is better, unless $\mathbf{NP} \subset \mathbf{DTIME}(n^{\log \log n})$ [4].

4.3 Iterative Improvement Algorithm

This is a variant of the popular EM algorithm, and it can be used to refine/improve existing solutions, e.g., solutions obtained with the previous algorithms. The algorithm iteratively improves the current models by fitting them more accurate in the existing segments, and then it finds a new segmentation given the improved models. The iteration continues until the error of the solution does not improve any more. The two steps of the iteration are the following.

Step 1: The current solution consists of a k-segmentation $(\mathbf{D}_1, \ldots, \mathbf{D}_k)$ of the input sequence and h models $\{M_1, \ldots, M_h\}$. Let $\overline{\mathbf{D}}_j$ be the set containing all the points in those segments that are assigned to the model M_t, $t = 1, \ldots, h$, that is $\overline{\mathbf{D}}_t = \{\mathbf{D}[i] \mid \mathbf{D}[i] \in \mathbf{D}_j \text{ and } \arg\min_l E(\mathbf{D}_j, M_l) = t\}$. Each model M_t is then replaced by M'_t, which is computed to be the optimal given all the points in $\overline{\mathbf{D}}_t$.
Step 2: Given the improved models M'_t a new k-segmentation is computed using a slight modification to the dynamic programming algorithm. In particular, the term $E(\mathbf{D}[j..i], M^*_{ji})$ is replaced by $\min_{t=1}^{h} E(\mathbf{D}[j..i], M'_t)$, that is, the best among the h given models M'_t is used for the subsequence $\mathbf{D}[j..i]$.

4.4 Putting Everything Together

Assume first that we want to segment the input sequence into k segments, using h predictive models, where the numbers k and h are given. Our algorithms starts by first segmenting the sequence into k segments, where each segment has its own predictive model. This segmentation task is performed using the dynamic programming algorithm, described in Section 4.1. We then treat the problem of reducing the number of models as a clustering problem, in which each of a k segments should be represented by one model from a smaller set of h models. We solve this task using the k-median or k-means algorithms, described in Section 4.2. Then we apply the iterative improvement algorithm described in Section 4.3 in order to refine the solution obtained in the previous steps.

Consider now the general case, in which the parameters k and h are not given, but need to be determined from the data. We address this model selection problem using the Bayesian information criterion (BIC). Since the error decreases when k and h increase, BIC suggests adding a penalty score to the error, which depends on the complexity of the model. Details on the BIC can be found in [9].

Under the Gaussian model with unit variance the BIC score is defined as $BIC = E + P\log(N)$, where E is the total sum of squares and P is the number of parameters. In our case, the number of parameters of the model is proportional to k and h. For our experiments we used $P = ck + dh$, where c and d are small integers that depend on the exact predictive model we use. For the model selection process, given upper bound values of k and h, we try all combinations of k segments with h models and we select the pair of parameters that yield the lowest BIC score.

Notice that if we use the facility location algorithm for clustering, we only have to iterate over the number of segments k. For each value of k the corresponding value of h that minimizes the BIC score is automatically selected by the facility location algorithm.

5 Experimental Data

A beautiful example of how this approach works in practice is provided by looking the task of predicting temperature based on solar radiation. While the application itself can be considered rather trivial, it nevertheless demonstrates how recurring models can be found when a segmental structure exists.

The motivation for this work originally stemmed from the desire to find recurrent models for explaining aerosol particle formation bursts in nature [14]. After repeated experiments we were forced to conclude that no such recurrent models with segmental structure exist and turn to other applications. We include a short description of this problem as in its way it is quite illustrative.

A somewhat different example is the haplotype prediction task: given d haplotypes and n markers, the task is to predict the marker value at a target haplotype. In theory this corresponds to reconstructing a haplotype in k parts, using only parts from h different ancestors.

5.1 Recurring Models in Environmental Data

The temperature prediction task. In nature various phenomena, such as temperature, follow a seasonal cycle. It seems reasonable to assume that different models might be needed to describe such phenomena in different seasons. Consider the problem of predicting the average daytime temperature based on solar radiation levels using linear regression. We expect different models to be valid in the summer, when high solar radiation days are the sunny and warm ones, and in the winter, when high solar radiation implies clear skies and cold temperatures.

We have searched for recurrent predictive models over three years of data consisting of daytime mean values of temperature and photosynthetically active radiation. Using the Bayesian information criterion (BIC) we can conclude that seven segments is an optimal choice, see Figure 1(a). This corresponds nicely with the fact that our data set runs over 4 winters and 3 summers. The models are summarized in Table 1. Note that the segmentation divides the time interval considered into winter and summer segments. Though not identical, the winter

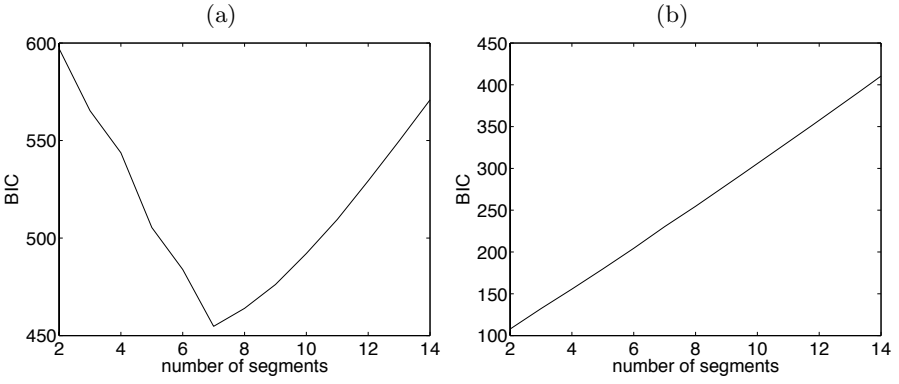


Fig. 1. The Bayesian information criterion (BIC) score as a function of the number of segments. (a) Temperature prediction task. (b) Aerosol particle formation prediction.

models resemble each other, as do the models valid in the summer segments. This is also apparent in Figure 2(a), which shows the error rate of the different models in a sliding window of width 40 days. It is evident, that there is a group of models which work well in the winter segments and another group of models for the summer segments. Two models will therefore probably be sufficient and we continue by looking for a good pair of models. Note, that the error can be arbitrarily large in the segments where the model is not valid.

Because the number of models we look for is small, we can do an exhaustive search. The chosen models are indicated in the third column in Table 1 and the fourth column gives the segments on which each of the two models is used. As a last step, the two models are refitted over the segments where they are valid. The final models are given in the fifth column.

Table 1. Temperature prediction task. The first column specifies the segment. The model valid in that segment is given in the next column. Since we are looking for recurring models, only two of these are chosen; these are indicated in the third column. Which one of these is used in the two model situation in each segment is indicated by the fourth column. The last column gives the winter and summer models, fitted over the corresponding intervals as given in the fourth column.

segment	model	chosen	used	refitted
2000/01/01 − 2000/04/16	$0.18R - 0.84$	W	W	$0.27R - 0.76$
2000/04/17 − 2000/12/16	$0.38R + 0.42$		S	
2000/12/17 − 2001/04/22	$0.16R - 0.85$		W	
2001/04/23 − 2001/10/19	$0.30R + 0.66$	S	S	$0.36R + 0.57$
2001/10/20 − 2002/04/21	$0.24R - 0.65$		W	
2002/04/22 − 2002/09/17	$0.18R + 0.89$		S	
2002/09/18 − 2002/12/31	$0.82R - 0.45$		W	

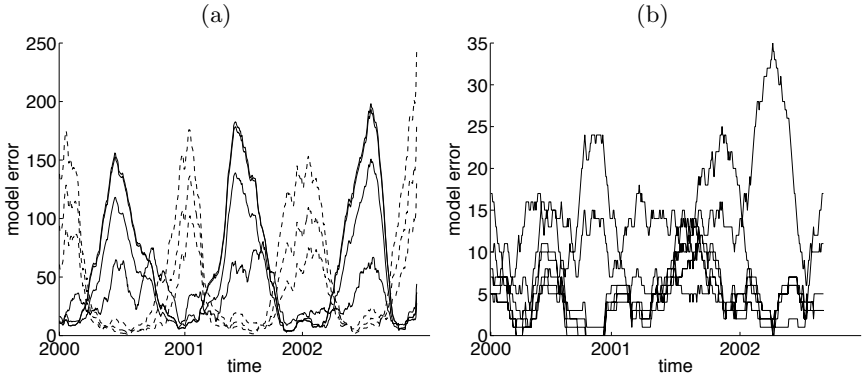


Fig. 2. The error for the (7,7) models in a sliding window of width 40 days. (a) Temperature prediction task. There are clearly two groups of models: one group which performs well on the winter segments and badly in the summer segments and another group with opposite performance behavior. (b) Aerosol particle formation prediction.

The cross-validated test errors for the optimal number of segments is shown in Table 2. Notice, that two models perform approximately as well as using a separate model for each segment, and is significantly better than using a single model for all days.

Aerosol particle formation. Another example of an environmental application is provided by the problem that was the original motivation of this work. In nature spontaneous bursts of aerosol particle formation occur frequently, and finding which factors are behind this phenomenon is an important question in atmospheric research [14]. Since this phenomenon has clear seasonal variation, it is reasonable to ask whether not only the occurrence pattern but also the causes behind the phenomenon would vary seasonally. Though the answer to this question turned out to be negative, we include this example to illustrate what happens when no segmental structure exists, as this case is also of great interest.

Our data consists of a set of meteorological measurements $\mathbf{X}$ and a target variable $\mathbf{y}$ telling us for each day t whether a particle formation burst occurred ($\mathbf{y}(t) = 1$) or not ($\mathbf{y}(t) = 0$). To keep this example simple we choose out of the various meteorological measurements only two variables known to be linked to particle formation bursts. We did much more extensive experiments with much the same results.

The cross-validated test errors for seven segments is shown in Table 2. This time there is no optimal choice of segments, see Figure 1(b), so we use this just to have one example of model performance. Notice, that now the segmented models perform no better than using a single model for the whole interval. Looking at Figure 2(b) we see no evidence of recurring models; if anything, it seems that some parts of the sequence are easier and some are more difficult to predict, even if segmentation in theory allows us to fit a particular model in a more difficult segment.

Table 2. Test scores for different choices of models. The given numbers are averages over 5-fold cross-validation. (a) Temperature prediction task. (b) Aerosol particle formation prediction. The score is defined as the sum of squared errors divided by the number of points for continuous variables (a) and as the number of misclassifications divided by the number of points for binary classification tasks (b).

models	(a)	(b)
different model for each segment	0.24	0.10
two models selected from the above	0.27	0.09
two models optimized over valid segments	0.27	0.10
single model for all days	0.59	0.10

5.2 The Haplotype Prediction Task

The haplotype prediction task is defined as follows: given d strings (haplotypes) of length n (markers), the task is to predict the value at a target string (haplotype). That is, we wish to represent a string using pieces from other strings. In theory, the (k, h)-restriction could be pretty efficient in this application: it means reconstructing a haplotype in k parts, using only parts from h different ancestors. In practice we of course do not have haplotype data from the ancestors, but from other descendants (hopefully) sharing the same ancestors.

In our experiments we used the Perlegen data set [11], which contains genotype data for 71 ethnically diverse individuals, including 23 African American samples, 24 European American samples and 24 Han Chinese samples. The original data

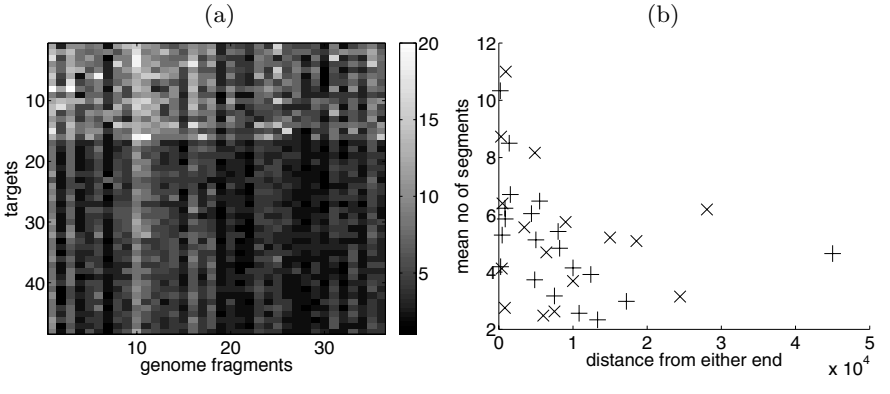


Fig. 3. (a) The number of segments needed to achieve an error rate of at most 3% for different haplotype fragments (x-axis) of 48 people (y-axis), picked from 3 different ethnic groups. On average, haplotypes from the African-American group (rows 1-16 from the top) are more difficult to predict using a segmental model than those from the European American (rows 17-32) or Han Chinese (rows 33-48) groups. Also, there are clearly locations which are either easier (columns 1 and 10) or more difficult (columns 19 and 30) to predict. (b) Average cost over targets as function of distance of fragment to beginning or end of the genome. Are fragments closer to either end more difficult to predict?

contains over 1.5 million SNPs (single nucleotide polymorphism), with an average distance between adjacent SNPs (or markers) of 1871 base pairs. Of this we used 36 haplotype fragments of length 500 markers picked at random from 4 different chromosomes. This means that on each of the 36 test runs our data consists of 142 binary valued haplotype segments of length 500. One haplotype at a time of a subset of 48 out of the 142 was kept as the target and the rest were used as models. The number of segments was selected so that the number of errors made in the modeling was at most 3%; the maximum number of segments allowed was 20. Since our models are fixed, no cross-validation was used. The length of the fragments is so short that not much recurrence can realistically be expected in this case. However, it is possible to gain insight into how different ethnic populations and different locations in the genome differ in the way ancestral segments of genomes are shared. Higher error rates imply shorter shared haplotypes and more local genomic or ancestral variation.

It is clear from the results, Figure 3, that haplotypes from the first group are more difficult to model, maybe implying more ancestral variation in the group. Also, the fragments in the beginning and the end of the haplotype seem more difficult to predict than those further away, but the sample size is really not large enough to draw any definite conclusions on this matter.

6 Conclusions

In this paper we considered the following problem: given a target sequence $\mathbf{y}$ and a sequence of observation values $\mathbf{X}$, find from a model class $\mathcal{M}$ a small collection of models that can be used to explain the target phenomenon in a piecewise fashion using the observation values.

We solved this problem by first using dynamic programming in order to find a good segmentation of the sequence into k segments. Each segment has its unique predictive model. We fix this segmentation, and select from the k models associated with these segments a smaller set of h models by transforming the problem into a k-median or a facility location problem. As a result we have a piecewise model, consisting of a segmentation of the sequence into k segments and a set of h models, one of which are used in each segment.

We have applied this approach to several real problems. In two of our examples we used environmental data to solve two different prediction tasks. Our third example described the task of constructing a target haplotype of the human genome using segments of other haplotypes, yet another very different prediction task. We demonstrated that when a segmental structure exists our method does find it with reasonable efficiency.

References

1. Bellman, R.: On the approximation of curves by line segments using dynamic programming. Communications of the ACM 4(6) (1961)
2. Bingham, E., et al.: Segmentation and dimensionality reduction. In: Jonker, W., Petković, M. (eds.) SDM 2006. LNCS, vol. 4165, Springer, Heidelberg (2006)

3. Charikar, M., Guha, S.: Improved combinatorial algorithms for the facility location and k-median problems. In: FOCS (1999)
4. Feige, U.: A threshold of ln n for approximating set cover. Journal of the ACM 45(4), 634–652 (1998)
5. Garey, M., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman (1979)
6. Gionis, A., Mannila, H.: Finding recurrent sources in sequences. In: RECOMB (2003)
7. Guha, S., Koudas, N., Shim, K.: Data-streams and histograms. In: STOC (2001)
8. Gwadera, R., Gionis, A., Mannila, H.: Optimal segmentation using tree models. In: ICDM (2006)
9. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. The MIT Press, Cambridge (2001)
10. Himberg, J., Korpiaho, K., Mannila, H., Tikanmaki, J., Toivonen, H.: Time series segmentation for context recognition in mobile devices. In: ICDM (2001)
11. Hinds, D.A., et al.: Wholegenome patterns of common DNA variation in three human populations. Science 307, 1072–1079 (2005)
12. Hochbaum, D.: Heuristics for the fixed cost median problem. Mathematical Programming 222, 148–162 (1982)
13. Hyafil, L., Rivest, R.L.: Constructing optimal binary decision trees is np-complete. Information Processing Letters 5, 15–17 (1976)
14. Hyvönen, S., et al.: A look at aerosol formation using data mining techniques. Atmospheric Chemistry and Physics 5, 3345–3356 (2005)
15. Keogh, E.J., et al.: Locally adaptive dimensionality reduction for indexing large time series databases. In: SIGMOD (2001)
16. Kleinberg, J.: Bursty and hierarchical structure in streams. In: KDD (2002)
17. Koivisto, M., et al.: An MDL method for finding haplotype blocks and for estimating the strength of haplotype block boundaries. In: PSB (2003)
18. Li, W.: DNA segmentation as a model selection process. In: RECOMB (2001)
19. Lin, J.-H., Vitter, J.S.: ϵ-approximations with minimum packing constraint violation. In: STOC (1992)
20. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77, 257–286 (1989)
21. Sarawagi, S.: Efficient inference on sequence segmentation models. In: ICML (2006)
22. Terzi, E., Tsaparas, P.: Efficient algorithms for sequence segmentation. In: SDM (2006)

Sequence Classification Using Statistical Pattern Recognition

José Antonio Iglesias, Agapito Ledezma, and Araceli Sanchis

Universidad Carlos III de Madrid,
Avda. de la Universidad, 30, 28911 Leganés (Madrid), Spain
{jiglesia,ledezma,masm}@inf.uc3m.es

Abstract. Sequence classification is a significant problem that arises in many different real-world applications. The purpose of a sequence classifier is to assign a class label to a given sequence. Also, to obtain the pattern that characterizes the sequence is usually very useful. In this paper, a technique to discover a pattern from a given sequence is presented followed by a general novel method to classify the sequence. This method considers mainly the dependencies among the neighbouring elements of a sequence. In order to evaluate this method, a UNIX command environment is presented, but the method is general enough to be applied to other environments.

Keywords: Sequence Classification, Sequence Learning, Statistical Pattern Recognition, Behavior Recognition.

1 Introduction

Sequential data mining is a broad discipline where the relationships of sequences of elements are used to different goals in different applications. A sequence is defined by the Merriam-Webster Dictionary as *a set of elements ordered so that they can be labelled with the positive integers*. Given a set of labelled training sequences, the main goal of a *sequence classifier* is to predict the class label for an unlabelled sequence. Furthermore, many other sequence learning tasks are considered: *sequence prediction* (given a set of sequences and one single sequence, predict the next item in the single sequence), *frequent subsequence discovery* (detect sub-sequences that occur frequent in a giving set of sequences), *sequence clustering* (cluster a set of unlabelled sequences in subsets), etc.

In particular, this paper focuses on the challenge of sequence classification. Let us define a sequence of n elements as $E = \{e_1, e_2,..., e_n\}$. Given a set of m classes $C = \{c_1, c_2,..., c_m\}$ we wish to determine which class $c_i \in C$ the sequence E belongs to. We present a novel method to classify a sequence.

This research is related to the framework used in the *RoboCup Coach Competition*. This competition of the Simulation League [1] was introduced in 2001, but changed recently in order to emphasize opponent-modelling approaches. The main goal of the current competition is to model the behavior of a team. A play pattern (way of playing soccer) is activated in a test team and the coach should detect this pattern and then, recognize the patterns followed by a team by observation.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 207–218, 2007.

We [2] presented a a very successful technique to compare agents behaviors based on learning the sequential coordinated behavior of teams. This technique was implemented by us in the 2006 Coach Team *Caos* [3].

In this paper, a sequence classifier is presented. As a sequence can represent an specific behavior, the classifier is evaluated in the environment of UNIX shell commands [4] in order to learn and classify a UNIX user profile.

The rest of the paper is organized as follows. In Section 2 we provide a brief overview of the related work on sequence classification. A summary of our approach is presented in section 3. Section 4 and 5 describe in detail the two parts of the proposed technique: Pattern Extraction and Classification. Experimental results are given in section 6. Finally, section 7 contains future work and concluding remarks.

2 Related Work on Sequence Classification

The main reason to need to handle sequential data is because of the observed data from some environments are inherently sequential.

An example of these environments is the DNA sequence. Ma et al. [5] present new techniques for bio-sequence classification. Given an unlabelled DNA sequence S, the goal in that research is to determine whether or not S is an specific promoter (a gene sequence that activates transcription). Also, a tool for DNA sequence classification is developed by Chirn et al. [6].

In the computer intrusion detection problem, Coull et al. [7] propose an algorithm that uses pair-wise sequence alignment to characterize similarity between sequences of commands. The algorithm produces an effective metric for distinguishing a legitimate user from a masquerader. In [8] Schonlau et al. investigate a number of statistical approaches for detecting masqueraders.

Another important reason to research sequential data is its motivation in the domain of user modelling. Bauer [9] present an approach towards the acquisition of plan decompositions from logged action sequences. In addition, Bauer [10] introduces a clustering algorithm that allows groups of *similar* sequences to be discovered and used for the generation of plan libraries.

In the area of agent modelling, Kaminka et al. [11] focus on the challenge of the unsupervised autonomous learning of the sequential behaviors of agents, from observations of their behavior. Their technique translates observations of a complex and continuous multi-variate world state into a time-series of recognized atomic behaviors. These time-series are then analyzed to find sub-sequences characterizing each agent behavior. In this same area, Riley and Veloso [12] present an approach to do adaptation which relies on classification of the current adversary into predefined adversary classes. This classification is implemented in the domain of simulated robotic soccer.

In Horman and Kaminka's work [13] a learner is presented with unlabelled sequential data, and must discover sequential patterns that characterize the data. Also, two popular approaches to such learning are evaluated: frequency-based methods [14] and statistical dependence methods [4].

3 Our Approach

In this work, the input consists of a set of sequences. A sequence is an ordered list of elements (events, commands,...) that represents an specific behavior (pattern). In the proposed framework, each sequence designates a class. The first part of this classifier is to discover and store the pattern (class) followed by each sequence. Then, a new small sequence is observed and compared to the stored patterns (classes) in order to determine which class it belongs to.

Therefore, the proposed approach has two main phases (Figure 1 shows an overview structure):

1. **Pattern Extraction.** A pattern can be defined as a compact and semantically sound representation of raw data (sequence). In our approach, every input sequence follows a different pattern, so a sequence pattern represents a class. Every sequence is pre-processed and represented in a special structure in order to get the pattern that it follows. This phase creates a library where the patterns obtained from each sequence are stored.
2. **Classification.** Once every pattern has been stored, a given sequence must be classified. The pattern of the given sequence is generated using the pattern extraction process. This pattern is then matched to every pattern in the *Our Patterns Library*.

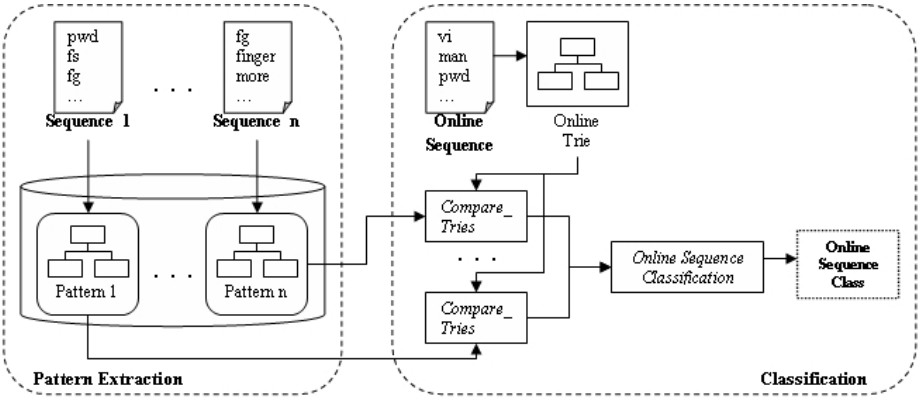


Fig. 1. Overview structure

4 Pattern Extraction

A sequence is an ordered list of elements that follows a pattern and it can be represented as $\{e_1 \rightarrow e_2 \rightarrow ... \rightarrow e_n\}$ where n is the length of the sequence. As a pattern represents a compact and semantically sound representation of the sequence, the first step is to extract from the sequence the elements more related

to it. Also, a pattern should be predictable, so we consider that the repeating elements of the sequences and its dependencies are related to the pattern.

Because of the previous supposition, in this work we propose the use of a trie data structure [15] [16] to store the useful sequence information. Therefore, the output of this first phase is a library in which the trie of each sequence is stored. As a trie represents the pattern followed by a sequence, this library is called *Pattern Library.*

4.1 Building a *Trie*

A trie (abbreviated from *retrieval*) is a kind of search tree similar to the data structure commonly used for page tables in virtual memory systems. This special search tree is used for storing strings in which there is one node for every common prefix and the strings are stored in extra leaf nodes.

The trie data structure has been used for retrieving a string efficiently from a set of strings; in [11] is used to learn a team behavior and in [17] to create frequent patterns in dynamic scenes. In this research we propose to use this data structure for a different goal: to store the main characteristics of a sequences in an effective way. The advantage of this kind of data structure is that every element is stored in the trie just once, in a way that each element has a number that indicates how many times it has been inserted on.

In the proposed trie structure, every element of the sequence is represented as a node. A path from the root to a node represents an ordered list of elements. Also, as the length of the sequences could be very long, the sequence must be split into smaller sub-sequences in order to store its elements in a trie. The length of these sub-sequences can modify both the size of the tries and the final results quite significantly.

The size of a trie depends on both the inserted and the repeated nodes. Due to the repeated nodes can vary with the sequence to treat; to analyze the relation between the sub-sequence length and the size of the generated trie, the number of nodes to insert is measured. Figure 2 shows the correlation between the sub-sequence length of a 100 elements sequence and the number of nodes (elements) to insert in the trie. As shown in Figure 2, given a sequence of n elements, to increment in one unit the length of the sub-sequence results in inserting $n/2$ elements in the trie. Therefore, the sub-sequence length is crucial in the proposed method.

Steps of Creating an Example *Trie.* An example of how to store a sequence in a trie data structure is shown as follows. In this example, a sequence consists of different words, which represent any kind of element. The sequence to insert into an initially empty trie is:

$\{w5 \rightarrow w1 \rightarrow w5 \rightarrow w1 \rightarrow w5 \rightarrow w3\}$

Firstly, this sequence must be split. Let 3 be the sub-sequence length, then the sequence is split in two sequences:

$\{w5 \rightarrow w1 \rightarrow w5\}$ and $\{w1 \rightarrow w5 \rightarrow w3\}$

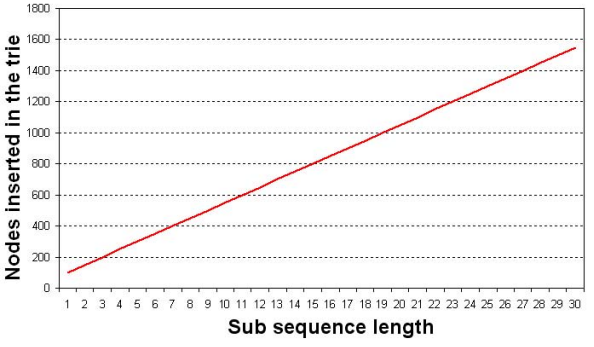


Fig. 2. Correlation between the sub-sequence length of a 100 elements sequence and the number of elements to insert in the trie

The first sequence is added as the first branch of the trie (Figure 3 A). Each element is labelled with the number 1 that indicates that the element has been inserted in the node once (in Figure 3, this number is enclosed in brackets). Because of repeating and significant sub-sequences are important to find the sequence pattern, the suffixes of the sub-sequences are also inserted. In the example, the suffixes $\{w1 \rightarrow w5\}$ and $\{w5\}$ are then added to the trie (Figure 3 B). Finally, after inserting the second sub-sequence and its remaining suffix, the complete trie is obtained (Figure 3 C).

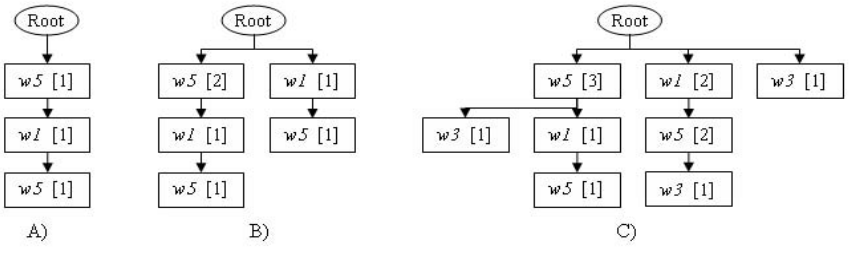


Fig. 3. Steps of creating an example trie

4.2 Evaluating Dependencies

In order to find the pattern that characterizes the elements of the sequence stored in a trie, two different approaches can be considered: frequency-based methods [14] and statistical dependence methods [4]. Considering the experimental results in [13], in this research, a statistical dependence method is used. In particular, to evaluate the relation between an element and its prefix (succession of elements previous to an element), we use one of the most popular statistic methods: the Chi-square test [18]. This statistical test enables to compare observed and expected element sequences objectively and evaluate whether

a deviation appears. Hence, every element (node) of a trie stores a value that determines whether an element is or not relevant with the previous ones.

To compute this test, it is necessary a 2x2 contingency table (also known as a cross-tabulation table). This table is filled with four frequency counters, as shown in Table 1. The counters are calculated as follows: The first number O_{11} indicates how many times the current element (node) is following its prefix. The number O_{12} indicates how many times the same prefix is followed by a different element. The number O_{21} indicates how many times a different prefix of the same length, is followed by the same element. The number O_{22} indicates how many times a different prefix of the same size, is followed by a different element.

Table 1. Contingency table

	Element	Different element	Total
Prefix	O_{11}	O_{12}	$O_{11} + O_{12}$
Different prefix	O_{21}	O_{22}	$O_{21} + O_{22}$
Total	$O_{11} + O_{21}$	$O_{12} + O_{22}$	$O_{11} + O_{12} + O_{21} + O_{22}$

The expected values are calculated as in Equation 1

$$Expected(E_{ij}) = \frac{(Row_i Total x Column_i Total)}{GrandTotal} \tag{1}$$

The formula to calculate chi-squared value, is given in equation 2

$$X^2 = \sum_{i=1}^{r} \sum_{j=1}^{k} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{2}$$

where: O_{ij} is the observed frequency and E_{ij} is the expected frequency.

This value is calculated for each element of the trie. Hence, the trie structure obtained in section 4.1. is modified to include this value in every node.

Finally, and as result of the first phase, every created trie (that represents a sequence pattern), is stored in the *Pattern Library*.

5 Classification

Given a new (and usually small) sequence to classify, the goal of this process is to determine which pattern (from the *Pattern Library*) the sequence is following. This process compares the given sequence to every pattern stored in the library. Therefore, the first step is to create the trie (that represents the pattern) corresponding to the sequence to classify. This trie (which we call *Testing Trie*) is generated using the process explained in section 4. This *Testing Trie* is then compared to every trie of the *Pattern Library*. Before describing the comparing algorithm, we should remember that every node in a trie is represented by: *Element* (word that indicates a specific element), *Prefix* (set of previous elements in the trie branch) and *Chi-Sq* (number that indicates the chi-square value for the node).

5.1 *Trie Sub-comparison*

If the *Testing Trie* and a trie from the *Pattern Library* (which we call *Class Trie*) represent the same pattern; the recurring elements to recurring prefixes should be similar in both tries. Accordingly, the key to compare two different tries is to note the possibility (measured by the chi-square value, *chi-sq*) that an element (e) occurs after a prefix (p) in both tries. In our method, the similarities and differences of two tries are represented by a set of *Trie sub-comparison* data structure, which can be defined as follow: *Trie sub-comparison* = (e, p, *comparison Value*) where the *comparison Value* represents the similarity or difference in both tries regarding the element e and its prefix p. This value is calculated from its chi-square values.

5.2 The Comparing Algorithm

The inputs of the algorithm presented below are the two tries to be compared. To apply this algorithm for a classifier method, it is executed once for every trie stored in *Pattern Library*. The number of executions is the number of classes (tries in library) and the two inputs are: the *Testing Trie* and a *Class Trie*.

The main points of the proposed comparing algorithm are the following:

For each node of the *Testing Trie*, its element and prefix are obtained. In the *Class Trie*, then a node with the same element and prefix is sought:

- If the present node is only in the *Testing Trie*:
  - It is interpreted as a difference between both tries. This difference together with the element and its prefix, are stored as part of the comparing result in the proposed structure *Trie sub-Comparison*. In this structure, the *comparison Value* indicates that there exists a difference between both tries. The *comparison Value* is the chi-square of the present node but its value is stored as a minus value because is representing a difference. As higher is the chi-square value, as more representative is the difference.
- If a node with the same element and prefix is in both tries:
  - The Chi-Square value of both nodes is compared: If the difference is lower than a threshold value, it means that there is some kind of similarity between the two tries. In this case, the *comparison Value* is the chi-square of the present node but it is stored as a positive value because is representing a similarity.

Figure 4 presents the basic structure of the proposed algorithm. The result of the algorithm is a set of *Trie Sub-Comparison* (*Comparison-Result*) that describes the similarities and differences of both tries. In this algorithm are used the following functions: *depthTrie(Trie T)*: returns the maximum depth of any of the leaves of the trie *T*. *getSetOfNodes(Level L, Trie T)*: returns a set nodes of the trie *T* in the level *L*. *getNode (Element E, Prefix P, SetOfNodes S)*: returns a node (from the set of nodes *S*) consisting of the element *E* and which prefix is *P*. (If a node with these parameters does not exist in *S*, the function returns

Algorithm 1. CompareSimilarityTries (*TestingTrie, ClassTrie*)

```
for level_i ← 2 to depthTrie (TestingTrie) do
    set_t ← getSetOfNodes(level_i, TestingTrie)
    set_c ← getSetOfNodes(level_i, ClassTrie)
    for all node_t in set_t do
        node_c ← getNode (element(node_t), prefix(node_t), set_c)
        if (node_c == null)   {the node is only in the Testing Trie}
            Trie-sub-Comparison ← Add(element(node_t), prefix(node_t), chi-sq (node_t)*-1)
            Comparison Result ← Add( Trie-sub-Comparison)
        else   {The node is in both tries}
        if (abs(chi-sq(node_t) - chi-sq (node_c)) ≤ ThresholdValue)
            Trie-sub-Comparison ← Add(element(node_t), prefix (node_t), chi-sq (node_t))
            Comparison-Result ← Add( Trie-sub-Comparison)
        end if
    end for
end for
```

Fig. 4. Basic Structure of the Comparing Algorithm of two tries

null). Finally, *element(node N)*, *prefix(node N)* and *chi-sq(node N)*: return the element, prefix and chi-square of the node *N*, respectively.

Once the *Testing Trie* has been compared with every *Class Trie*; we add up the *comparison Value* for every *Trie Sub-Comparison* obtaining an amount for each *Class Trie*. This amount represents the similarity between the given sequence and the class. Therefore, the result of the classifier is the *Class Trie* with a higher value (positive values represent similarity).

6 Experimental Setups and Results

In order to evaluate the proposed method, we have fully implemented a system that classifies UNIX command line sequences. In this environment, we extract the profile of a user from its UNIX commands sequences and then we classify a given sequence in one of the user profile previously stored. This task is very useful in computer intrusion detection.

We used 9 sets of sanitized user data drawn from the command histories of 8 UNIX computer users at Purdue University over 2 years [19]. The data is drawn from *tcsh* history files and has been parsed and sanitized to remove filenames, user names, directory structures, etc. Command names, flags, and shell metacharacters have been preserved. Additionally, tokens have been inserted to divide different user sessions. Also, and it is very important in our research, tokens appear order issued within the shell session, but no timestamps are included in the data.

For evaluating the proposed method, we have used the benchmark data sets from [19]. Each input file contains from about 10.000 to 60.000 commands. Firstly, for each user is created a trie (user profile library) that represents its

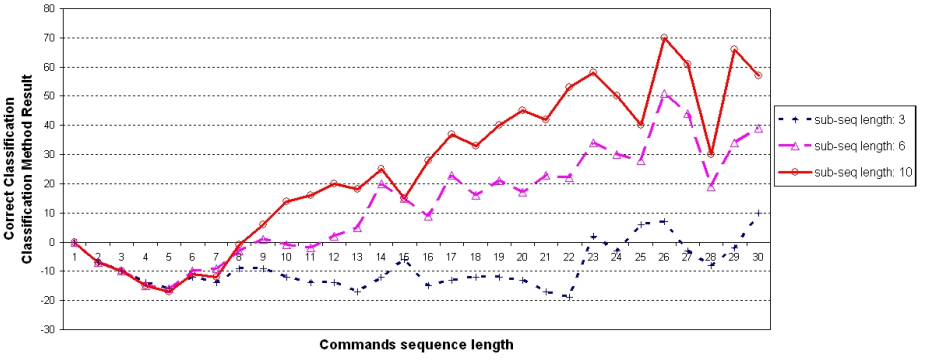


Fig. 5. Comparing Results. Unix Commands Classification - User 6.

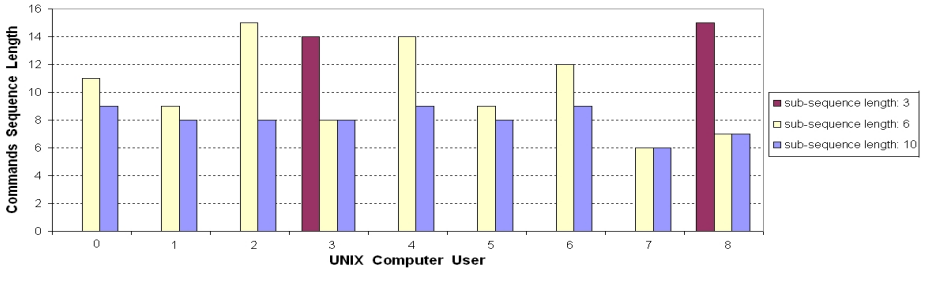


Fig. 6. Length necessary to classify a UNIX computer user correctly

behavior. As we explained in section 4, the length of the sub-sequences can modify both the size of the tries and the final results quite significantly, so we have executed our method with different sub-sequences length in order to evaluate the results.

Once tries of different sizes have been built for each user, we conducted extensive experiments. To evaluate our method we use the user profile library (set of classes) and a given sequence to be classified (this sequence is labelled because it is obtained from a user file). As we want to recognize a user as soon as possible, we classify sequences of very different sizes. After using the proposed technique, a comparing value is obtained for each user profile (*class value*). From these results, the given sequence is classified in the class with the highest value.

In order to evaluate the result and represent them graphically, for each given sequence we calculate a result value. This value represents the difference between the value obtained for the given sequence class and the highest *class value*:

– If the obtained value is negative, it means that there is another class considered by our method more similar to the given sequence (our classification is wrong).

– If the value is zero, the classification is right. Also, to evaluate the correctness of the result and the efficiency of the algorithm, the following value is calculated: difference between the obtained value and the second highest value. Therefore, as higher is this calculated value, as better is the classification.

Figure 5 shows the results for a sequence obtained from the user 6 commands file. The X-axis represents the given sequence length. The Y-axis represents the calculated value to evaluate our method. In the graph, three different subsequence lengths (3, 6 and 10) are represented. Because of the result of the method can depend on the given sequence, each point represented in the graph is the average value of 25 different tests conducted. As we can see, the best result is obtained using long sub-sequences. However, the size of the trie and the time consuming to build the trie and classify the sequence are highly increased with the length of the sub-sequences.

Because of lack of space, we have omitted the graphs that represent the result for the other 8 users. However, these results are also successful and the representative values are similar. Considering the results, we obtain: Let 6 be the sub-sequence length, then this method is able to correctly classify every given sequence of more than 15 commands.

Figure 6 represents the length of the given sequence necessary to classify correctly one of the 9 evaluated users. Considering a sub-sequence length of 3, the classification is not usually correct after even 50 commands. Only two users (3 and 8) are correctly classified with this size.

7 Conclusions and Future Work

In Horman and Kaminka's work [13] a learner to discover sequential patterns is presented. Also, to overcome the length bias obstacle, they normalize candidate pattern ranks based on their length. To improve the results in our research, a normalization method for comparing tries of different lengths could be implemented.

Previous to this research, we have developed a method for comparing agents behavior. The method was based on learning the sequential coordinated behavior of teams. The result of that method was successfully evaluated in the *RoboCup Coach Competition*. Related to that research, in this paper a sequence classification using statistical pattern recognition is presented. This method consists of two different phases: Pattern extraction and Classification. The goal of the first phase (in which previous works have been considered) is to extract a pattern or behavior from a sequence. The extracted pattern is represented in a special structure: *trie*. The second phase presents a method to compare different patterns (*tries*) in order to classify a given sequence.

In order to evaluate the proposed technique in a specific environment, we focus our experiments on the task of classify UNIX command line sequences. The technique was evaluated in a rigorous set of experiments and the results demonstrate that it is very effective in such tasks.

On the other hand, other approaches have been applied in the environment presented in this paper (UNIX shell commands): using *Hidden Markov Models* (HMMs) [20] [21] or employing instance-based learning (IBL) [20]. However, the goals to achieve by these methods differ from our proposal. A detailed analysis confronting our approach with others is proposed as future work.

Acknowledgments. This work has been supported by the Spanish Government under project TRA2004-07441-C03-2/IA.

References

1. The robocup 2005 coach competition web page (December 2006), http://staff.science.uva.nl/~jellekok/robocup/rc05
2. Iglesias, J.A., Ledezma, A., Sanchis, A.: A comparing method of two team behaviours in the simulation coach competition. In: Torra, V., Narukawa, Y., Valls, A., Domingo-Ferrer, J. (eds.) MDAI 2006. LNCS (LNAI), vol. 3885, pp. 117–128. Springer, Heidelberg (2006)
3. Iglesias, J.A., Ledezma, A., Sanchis, A.: Caos online coach 2006 team description. In: CD RoboCup 2006, Bremen, Germany (2006)
4. Howe, A.E., Cohen, P.R.: Understanding planner behavior. Artificial Intelligence 76(1-2), 125–166 (1995)
5. Ma, Q., Wang, J.T.-L., Shasha, D., Wu, C.H.: Dna sequence classification via an expectation maximization algorithm and neural networks: a case study. IEEE Transactions on Systems, Man, and Cybernetics, Part C 31(4), 468–475 (2001)
6. Chirn, G.-W., Wang, J.T.-L., Wang, Z.: Scientific data classification: A case study. ICTAI '97: Proceedings of the 9th International Conference on Tools with Artificial Intelligence , 216 (1997)
7. Coull, S.E., Branch, J.W., Szymanski, B.K., Breimer, E.: Intrusion detection: A bioinformatics approach. In: Omondi, A.R., Sedukhin, S. (eds.) ACSAC 2003. LNCS, vol. 2823, pp. 24–33. Springer, Heidelberg (2003)
8. Schonlau, M., DuMouchel, W., Ju, W., Karr, A., Theus, M., Vardi, Y.: Computer intrusion: Detecting masquerades, statistical Science (2001) (submitted)
9. Bauer, M.: Towards the automatic acquisition of plan libraries. ECAI , 484–488 (1998)
10. Bauer, M.: From interaction data to plan libraries: A clustering approach. In: IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, pp. 962–967. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1999)
11. Kaminka, G.A., Fidanboylu, M., Chang, A., Veloso, M.M.: Learning the sequential coordinated behavior of teams from observations. In: Kaminka, G.A., Lima, P.U., Rojas, R. (eds.) RoboCup 2002. LNCS (LNAI), vol. 2752, pp. 111–125. Springer, Heidelberg (2003)
12. Riley, P., Veloso, M.M.: On behavior classification in adversarial environments. In: Parker, L.E., Bekey, G.A., Barhen, J. (eds.) DARS, pp. 371–380. Springer, Heidelberg (2000)
13. Horman, Y., Kaminka, G.A.: Removing statistical biases in unsupervised sequence learning. In: IDA, 2005, pp. 157–167 (2005)
14. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Yu, P.S., Chen, A.S.P. (eds.) Eleventh International Conference on Data Engineering, pp. 3–14. IEEE Computer Society Press, Taipei, Taiwan (1995)

15. Fredkin, E.: Trie memory. Comm. A.C.M. 3(9), 490–499 (1960)
16. Knuth,: The Art of Computer Programming, vol. 3. Addison-Wesley, Reading (1973)
17. Huang, Z., Yang, Y., Chen, X.: An approach to plan recognition and retrieval for multi-agent systems. In: Proceedings of AORC (2003)
18. Chiang, C.L.: Statistical Methods of Analysis, World Scientific, Suite 202, 1050 Main Street, River Edge, NJ 07661 (2003)
19. Newman, C.B.D.J., Hettich, S., Merz, C.: UCI repository of machine learning databases (1998), http://www.ics.uci.edu/$\sim$mlearn/MLRepository.html
20. Lane, T., Brodley, C.E.: An empirical study of two approaches to sequence learning for anomaly detection. Mach. Learn. 51(1), 73–107 (2003)
21. Yeung, D.-Y., Ding, Y.: Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognition 36(1), 229–243 (2003)

Subrule Analysis and the Frequency-Confidence Diagram

Jürgen Paetz

63179 Obertshausen, Germany
juergen.paetz@t-online.de
www.appliedsoftcomputing.eu

Abstract. The quality of single classification rules for numerical data can be evaluated by different measures. Common measures are the frequency and the confidence of rules beside others. A problem with these measures is that they are valid for a rule only if an uniform distribution of the data, corresponding to the rule, is assumed. Since this is usually not the case, especially when considering high dimensional data, subrules and their properties should be considered additionally. The frequency and the confidence values of the subrules, summarized in a diagram, give more information about the quality of the rules than the properties of the rules solely.

Keywords: Rules, subrules, frequency, confidence.

1 Introduction

Rules for numerical data are usually described by interval conditions. For each dimension i a condition C_i is formulated in the format: $x_i \in [a_i, b_i]$, $x_i, a_i, b_i \in \mathbb{IR}$, $i = 1, \ldots, k$. The rule R is then formulated as: 'if C_1 and C_2 and … and C_k then P' with a property P for the data like a class label for example. In this common case the conditions of a rule R correspond to a rectangle $\boldsymbol{R} = [a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_{k-1}, b_{k-1}] \times [a_k, b_k]$. Modifications are closed or open intervals or infinite intervals. Less than k conditions can be considered in order to omit irrelevant dimensions. For an introduction to rules and related topics see [1], [2] for example. If a specific precision of the data should be considered, like one decimal after comma, the numbers x_i, a_i, b_i are elements of an exact space $\mathbb{IIR}$, where only such numbers are allowed [3]. More general other parametrized geometric figures than rectangles could be considered like ellipses or triangles, although in the literature the term rule is used in connection with rectangles. However, for our considerations it makes no difference what kind of these numerical rules is considered.

Usually, one is interested in a property P of a rule R or $\boldsymbol{R}$. Important values for rating the quality or the interestingness of a rule are its frequency freq(R) and its confidence conf(R). The frequency is the proportion of the data, that are elements of $\boldsymbol{R}$, related to the number of all the data samples in the dataset D, i.e., $|\boldsymbol{R}| \,/\, |D|$. The confidence is defined as the proportion of the elements in $\boldsymbol{R}$ that fulfill the property P, related to all the samples in $\boldsymbol{R}$, i.e., $|P(\boldsymbol{R})| \,/\, |\boldsymbol{R}|$. In the case that no elements are located

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 219–228, 2007.

in $\boldsymbol{R}$ the confidence is not defined. Commonly, the values are multiplied by 100 in order to have percentages. These measures were originally defined for association rule learning with symbolic data [4]. Although there are relations between symbolic and numerical data analysis for classification, we concentrate on numerical data analysis in this contribution. For further reading on association rules refer to [5] for example.

Although such measures like frequency and confidence can be defined for a rule $\boldsymbol{R}$ formally, they are statistically valid or useful only if an uniform distribution of the data is assumed. Otherwise, different regions $\boldsymbol{S} \subset \boldsymbol{R}$ might have significantly different frequency or confidence values, so that freq($\boldsymbol{R}$) and especially conf($\boldsymbol{R}$) is not a useful characterization of $\boldsymbol{R}$ and what is worse, the values are misleading. Numerical rule learners are often based on heuristics to reduce run time complexity, and the generation of inhomogeneous rules is not explicitly forbidden, e.g. [6] (decision trees), [7] (support cuts of fuzzy rules), [8] (rules from data streams). In the next section we discuss several examples where the values of a rule $\boldsymbol{R}$ give not a satisfying characterization of it.

For a better description of a (rectangular) rule all subrules, i.e., all (rectangular) subregions, $\boldsymbol{S} \subset \boldsymbol{R}$, need to be considered with their values. For $\mathbb{R}$ this would be an infinite number of rules, and even for an exact space $\mathbb{IR}$ the number would be high due to combinatorial explosion. In Section 2 we propose the random generation of subrules $\boldsymbol{S}_j$ in order to obtain pairs (freq($\boldsymbol{S}_j$), conf($\boldsymbol{S}_j$)). These pairs are noted in a diagram, that allows for a more meaningful view on the rule $\boldsymbol{R}$. Some examples of such diagrams and their interpretation are given in Section 3.

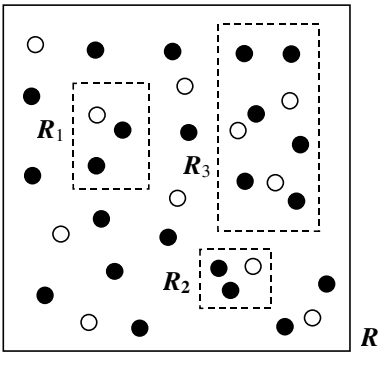


Fig. 1. A rule $\boldsymbol{R}$ and three subrules $\boldsymbol{R}_1$, $\boldsymbol{R}_2$, and $\boldsymbol{R}_3$. With respect to the property P = 'the class of the sample is black,' the confidence of the rules is equal. It is conf($\boldsymbol{R}$) = conf($\boldsymbol{R}_1$) = conf($\boldsymbol{R}_2$) = conf($\boldsymbol{R}_3$) = 66.67%.

2 Problems with Rule Quality Measures and Subrules

Two numerical rules can be compared by rule quality measures, whether they have been generated by one method or by two different ones. In Fig. 1 a rule $\boldsymbol{R}$ is depicted with conf($\boldsymbol{R}$) = 66.67% (rounded to two decimals). Without loss of generality its frequency is assumed to be 100% (33 samples), but could be any other value as well.

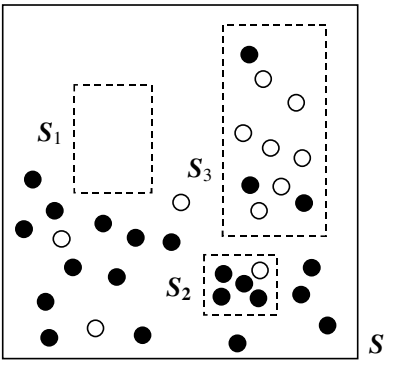


Fig. 2. A rule S and three subrules S_1, S_2, and S_3. With respect to the property P = 'the class of the sample is black,' the confidence of the rules is conf(S) = 66.67%, conf(S_1) not defined, conf(S_2) = 80.00%, and conf(S_3) = 30.00%.

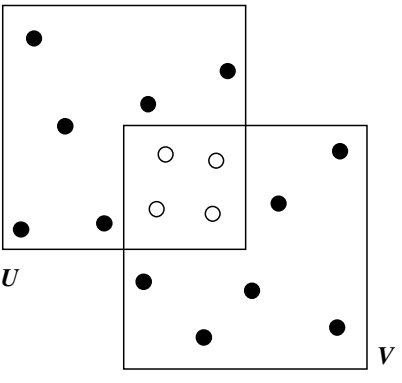


Fig. 3. Rules U and V, both with a higher confidence for class black. In the overlapping region only samples of class white are located.

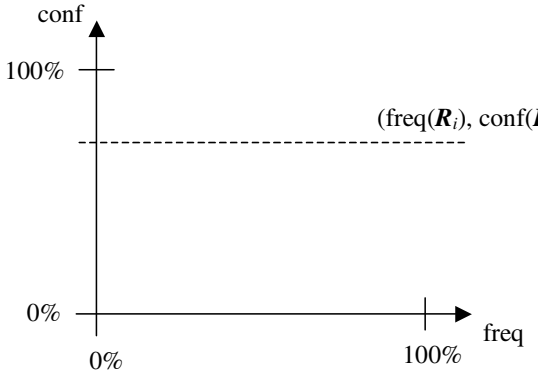


Fig. 4. In the ideal case for all frequencies > 0% of subrules R_i the confidence remains the same

The subrules $\boldsymbol{R}_1$, $\boldsymbol{R}_2$, and $\boldsymbol{R}_3$ have the same confidence with frequency freq($\boldsymbol{R}_1$) = freq($\boldsymbol{R}_2$) = 9.09% and freq($\boldsymbol{R}_3$) = 27.27%. In this more homogeneous rule the prediction of a new sample as being black can be assumed to be actually about 66.67%. In Fig. 2 the rule $\boldsymbol{S}$ has the same frequency and confidence values as the rule $\boldsymbol{R}$ in Fig. 1. But the subrules have different values: freq($\boldsymbol{S}_1$) = 0.00%, freq($\boldsymbol{S}_2$) = 15.15%, freq($\boldsymbol{S}_3$) = 30.30%, conf($\boldsymbol{S}_1$) not defined, conf($\boldsymbol{S}_2$) = 80.00%, and conf($\boldsymbol{S}_3$) = 30.00%. For a new sample it is a vague hypothesis to state that it is of class 'black' with a probability of 0.67. This is due to the non-uniformity of the rule $\boldsymbol{S}$, what is not expressed in the frequency and confidence values. Compared to rule $\boldsymbol{R}$, the rule quality of $\boldsymbol{S}$ is intuitively lower. Since high dimensional data is likely to tend to more extreme distributions with empty regions, the rules should be analysed more carefully.

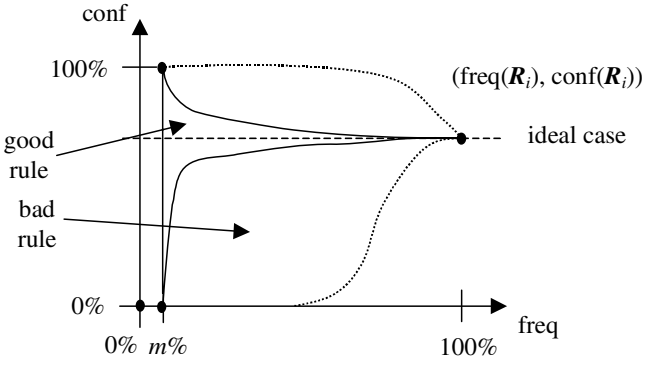


Fig. 5. The characteristic areas of the frequency-confidence pairs that appear for good rules (area limited by solid lines, case A) and bad rules (area limited by dotted lines, case B). The pairs are elements of the area including the borders. The *m*% is the frequency for one sample. Empty subrules can be marked formally as (0%,0%).

The following problems are a consequence of inhomogeneous rules: less interpretability, random classification of unknown data, wrong decisions with overlapping rules. In Fig. 1 the rule $\boldsymbol{R}$ can be clearly interpreted as a rule, that mainly represents samples of class black. Wherever a new, unknown sample is located in the rule $\boldsymbol{R}$, it can be classified as being black. Using the rule $\boldsymbol{S}$ in Fig. 2 it depends on the location of the sample in $\boldsymbol{S}$ how it should be classified in the best way. If the sample is located in $\boldsymbol{S}_1 \subset \boldsymbol{S}$, then it might be a better choice to classify the sample as 'unknown,' since the properties of the rule $\boldsymbol{S}$ are not based on data located in the subregion $\boldsymbol{S}_1$. If the overlapping of two inhomogeneous rules is used for a class decision, then the decisions could be wrong. In Fig. 3 two inhomogeneous rules are depicted, both with a higher confidence for class black than for class white. Although one would intuitively assume, that in the overlapping region the decision of a sample as being black is supported by the two rules, this assumption is wrong due to the inhomogeneity of the rules. In fact, in the overlapping region all samples are white.

We propose the random generation of subrules in order to check the homogeneity or uniformity of a rule. The frequency and confidence values are depicted in a

diagram with an additional counter for empty regions. In Fig. 4 the ideal case is depicted, where the data is uniformly distributed and where all the samples are available with an infinite precision. In Fig. 5 (case A) a case is depicted where the data is almost uniformly distributed, and where the samples are available with a predefined finite precision. Of course, if the intervals of the subrule would become smaller than the precision, then empty subrules are generated. This is not a problem of the rule itself. If the rules are very small with respect to the precision, it is possible that only one sample is element of a subrule, resulting in 0,00% or 100,00% confidence. In Fig. 5 (case B) the worst case is depicted, where even large subrules have already very different confidence values. It is clear that no positive frequency can be lower than the frequency $m\%$ of a rule with a single sample. The empty subrules can be counted formally as (0%,0%)-pairs. Good rules should have less empty subrules than bad rules. The prediction of samples, that are located mostly in empty subrules, is less significant than of those, that are located mostly in confident subrules.

To quantify the homogeneity with the frequency-confidence diagram we define firstly the (formal) fc-area.

Definition 1
Let there be a dataset D and a rule R with corresponding region $\boldsymbol{R}$. Generate the set $\boldsymbol{R}_{\text{sub}}$ of all r subrules $\{\boldsymbol{R}_1,\ldots,\boldsymbol{R}_r\}$. For every subrule out of $\boldsymbol{R}_{\text{sub}}$ calculate the frequency and confidence. The area a($\boldsymbol{R}$), that includes all frequency-confidence pairs, is normed by the area a(H), $H = [m\%, 100\%] \times [0\%, 100\%]$, so that a($\boldsymbol{R}$) $\in$ [0, 1]. The (normed) area a($\boldsymbol{R}$) is called the (formal) *fc-area* of $\boldsymbol{R}$.

In the ideal case the fc-area of a rule is 0. For a homogeneous rule it is nearer to 0 than to 1. For an inhomogeneous rule it is clearly greater than 0. If real numbers are allowed, the number of subrules is infinite. Even with a discrete number of possible values in each dimension, the number of subrules is combinatorially exploding. Subsequently, Definition 1 is only useful for theoretical considerations or on rule examples with low complexity. The combinatorial explosion of subrules leads to the situation that many randomly generated subrules are empty, i.e., the frequency of the subrules is 0. Let us consider the Cancer dataset for example [9]. In each dimension the minimum and maximum values are 0.1 and 1.0, respectively. The appearing numbers in the Cancer dataset are {0.1, 0.2, 0.3, …, 0.8, 0.9, 1.0} only. With 9 dimensions the Cancer data space S_{Cancer} could theoretically contain 10^9 elements. In fact it only has 699 samples, what is $6.99 \cdot 10^{-5}\%$ of S_{Cancer}. The rule $\boldsymbol{C}$, defined in Section 3, covers theoretically 75264000 samples, and actually 134 samples ($1.78 \cdot 10^{-4}\%$). So it is not useful to generate a very high number of mostly empty subrules what is due to the small percentages of actually covered positions with respect to the theoretically possible ones.

A first idea was to allow only the generation of subrules with a certain percentage of volume in each dimension. But even with high percentages > 90% the randomly generated subrules remained mostly empty with no significant frequency-confidence diagram. It turned out that the restriction to subrules is useful, that remain equal in all dimensions with the exception of one randomly selected dimension. We will see in

input: rule $\boldsymbol{R}$
output: fc-area a($\boldsymbol{R}$)

calculate (freq($\boldsymbol{R}$),conf($\boldsymbol{R}$)) on the data;
calculate the minimum frequency m on the data;
generate s subrules $\boldsymbol{S}_j$ of $\boldsymbol{R}$ by generating randomly a subinterval …
    … in one dimension;
for all subrules $\boldsymbol{S}_j$ **do**
  calculate (freq($\boldsymbol{S}_j$),conf($\boldsymbol{S}_j$)) on the data;
  % (only a check of the data in $\boldsymbol{R}$ is necessary, the other data is not in $\boldsymbol{S}_j$)
end
P := {(freq($\boldsymbol{S}_j$),conf($\boldsymbol{S}_j$))}$_{j=1,\ldots,s}$;
$Q := P \cup$ {(freq($\boldsymbol{R}$),conf($\boldsymbol{R}$)),(m,0),(m,100)};
determine estimated, dimensionally restricted, normed fc-area a($\boldsymbol{R}$) …
    … as approximative integral area a(Q)/a(H);

Fig. 6. Calculation of the fc-area of a rule $\boldsymbol{R}$

Section 3 that subrules lead to significant frequency-confidence diagrams, when the subrule property is considered in each dimension. To be correct, we modify Definition 1 slightly.

Definition 1'
Let there be a dataset D and a rule R with corresponding region $\boldsymbol{R}$. Generate a set $\boldsymbol{R}_{\text{sub}}$ of s subrules {$\boldsymbol{R}_1,\ldots,\boldsymbol{R}_s$}, where only in one dimension a subinterval is considered for each subrule. The number s of subrules should be chosen large enough, so that the estimation of the fc-area becomes stable. For every subrule out of $\boldsymbol{R}_{\text{sub}}$ calculate the frequency and confidence. The normed area a($\boldsymbol{R}$), that includes all frequency-confidence pairs, is called the (estimated, dimensionally restricted) *fc-area* of $\boldsymbol{R}$.

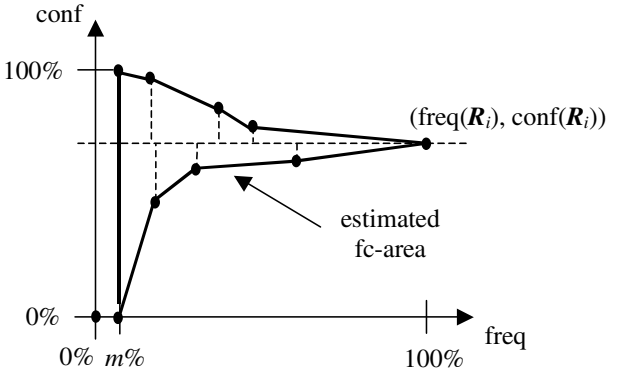


Fig. 7. The frequency-confidence pairs are linearly interpolated. The area of the resulting trapezoids is summed up to the estimated (unnormalized) fc-area.

To avoid a high statistical bias due to the chosen number s of subrules, a set of increasing numbers can be used for the estimation of the fc-area. If the further increasing of s does not lead to a significantly different fc-area, then s is a suitable parameter. In Fig. 6 we summarize the practical subrule analysis with the estimated, dimensionally restricted fc-area.

The approximative integration can be done by calculating the area of the piecewise linearly interpolated points in Q, differentiating the points with higher and lower values than conf($\boldsymbol{R}$), cf. Fig. 7. We remark that this area is not the same as the area of the convex hull of Q, that is not used here, because the set Q is not convex.

In ROC analysis a diagram is plotted with specifity and sensitivity pairs of a classifier [10]. The area under the ROC curve (AUC) is a performance measure for a classifier, a set of rules for example. The fc-area measures the homogeneity of a single rule. While AUC can measure the classification accuracy of any classifier, with the fc-area tolerances of the statistical properties of a rule are quantified.

3 Examples on Real World Datasets

In this section the (estimated, dimensionally restricted) fc-areas of two rules are determined for the datasets Diabetes (2, 8, 768) and Cancer (2, 9, 699) [9], respectively. In brackets the number of classes, the dimensionality, and the number of samples is noted. For our experiments we use rules that we have generated earlier. We utilized s = 1000 randomly generated subrules for every rule. For the Cancer dataset the following rule $\boldsymbol{C}$ is analysed as an example:

$\boldsymbol{C}$) **if** var 1 ∈ [0.50, 1.00] **and** var 2 ∈ [0.40, 1.00] **and** var 3 ∈ [0.30, 1.00] **and** var 4 ∈ [0.10, 1.00] **and** var 5 ∈ [0.20, 0.80] **and** var 6 ∈ [0.10, 1.00] **and** var 7 ∈ [0.30, 1.00] **and** var 8 ∈ [0.10, 1.00] **and** var 9 ∈ [0.10, 0.40] **then** class 1 with freq = 19.2% and conf = 94.0%.

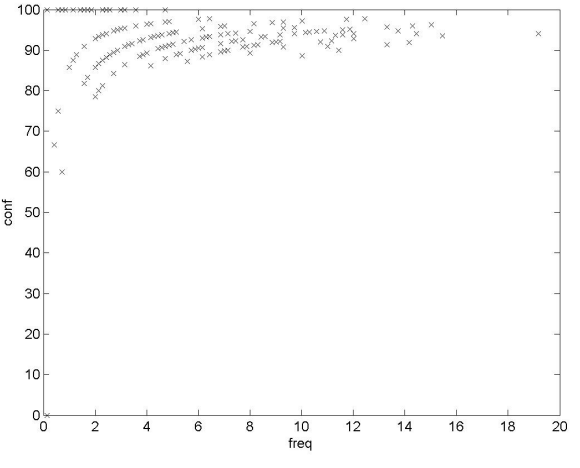


Fig. 8. Frequency-confidence diagram for the rule $\boldsymbol{C}$

In Fig. 8 the frequency-confidence diagram for rule ***C*** is depicted. There are subrules with a frequency less than 6% that have 100% confidence. There is a variety of subrules with a frequency of 0% to 16% with a confidence of 60% to 100%. The (normed) fc-area is about 0.06. A number of 159 subrules out of 1000 have a frequency equal to 0. Due to the low fc-area the rule ***C*** can be considered as a good rule. The rule ***D*** for the Diabetes dataset is as follows:

D) **if** var 1 ∈ [3.00, 10.00] **and** var 2 ∈ [84.00, 165.00] **and** var 3 ∈ [0.00, 72.00] **and** var 4 ∈ [0.00, 32.00] **and** var 5 ∈ [0.00, 168.00] **and** var 6 ∈ [0.00, 37.20] **and** var 7 ∈ [0.09, 0.63] **and** var 8 ∈ [22.00, 55.00] **then** class 1 with freq = 10.9% and conf = 71.4%.

The frequency-confidence diagram for rule ***D*** is depicted in Fig. 9. With 142 out of 1000 rules being empty, the fc-area is about 0.18. With lower frequencies we notice a larger spread of confidence values from 20% to 100%. In Fig. 9 all frequency-confidence pairs with subintervals in the first dimension are marked with a circle instead of a cross. These pairs are distributed like the others. The same holds for the other dimensions. Overall, no specific irregularities can be noticed for any dimension.

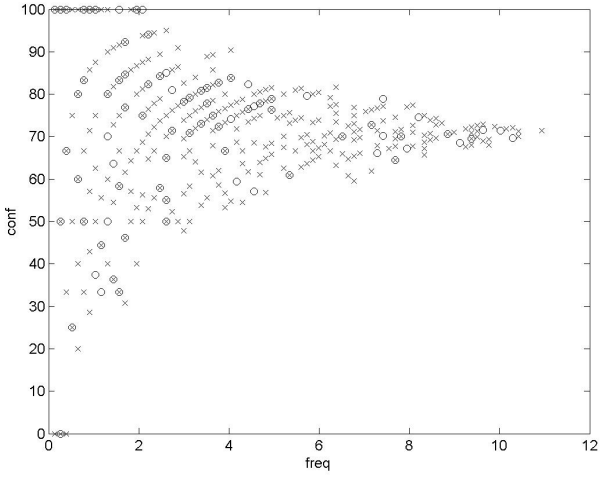


Fig. 9. Frequency-confidence diagram for the rule ***D***. The pairs stemming from subrules with subintervals in the first dimension are marked with a circle instead of a cross.

Another visualization could be made for fuzzy rules, where the subrules between the core region and the support region for specific cuts between 0 and 1 can be highlighted. This is not considered here in detail, but the pair corresponding to the core region is more likely located in the upper left corner, and the pair corresponding to the support region is located on the right side.

4 Conclusion

In this contribution we have demonstrated the drawbacks of two common rule measures. Since an uniform distribution of the data is usually not present, the rule measures are not representative for the whole rule region. Although they should state more precisely the properties of a rule, they might even be misleading without any further analysis. A deeper analysis can be performed by the consideration of subrules. The frequency and confidence values of subrules can be depicted in a diagram, that summarizes the properties of a rule more accurately. It can be seen as a homogeneity property of a rule. If all or a representative number of subrules have similar properties, then the rule $\boldsymbol{R}$ is homogeneous and it is a good rule from this viewpoint. If the subrules have very different values, than it is better not to consider $\boldsymbol{R}$ as a good rule. In this case the rule $\boldsymbol{R}$ would be too inhomogeneous to allow for accurate predictions. A split of $\boldsymbol{R}$ in different subrules might then lead to more accurate rules. It was discussed that inhomogeneous rules are less interpretable, less suitable as part of classifiers, and less accurate with regard to unknown data. Especially when errors are costly, a subrule analysis can prevent for undesired costs. Since a formal subrule analysis is computationally costly itself, we restricted a practicable analysis to dimensionally restricted subrules with a frequency-confidence diagram as visualization. The fc-area helps to decide whether the statistical properties frequency and confidence are valid properties of the rule or not.

Further work could be concerned with the optimization of the geometric shape of a rule $\boldsymbol{R}$ and its subrules with respect to the frequency-confidence diagram. A comparison of complete rule sets of different rule classifiers by a subrule analysis can be done.

References

1. Berthold, M.R., Hand, D.J.: Intelligent Data Analysis: An Introduction, 2nd edn. Springer, Berlin (2003)
2. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press, Cambridge, MA (2001)
3. Paetz, J.: On the Role of Numerical Preciseness for Generalization, Classification, Type-1, and Type-2 Fuzziness. In: Proc. of the 1st IEEE Int. Symp. on Foundations of Computational Intelligence (Symposium Series on Computational Intelligence), Honolulu, HI, USA, pp. 208–213. IEEE Computer Society Press, Los Alamitos (2007)
4. Agrawal, R., Skrikant, R.: Fast Algorithms for Mining Association Rules. In: Proc. of the 20th Int. Conf. on Very Large Databases, Santiago de Chile, Chile, pp. 487–499 (1994)
5. Zaki, M.J.: Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. IEEE Transactions on Knowledge and Data Engineering 17, 462–478 (2005)
6. Rokach, L., Maimon, O.: Top-Down Induction of Decision Trees Classifiers - A Survey. IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews 35, 476–487 (2005)
7. Huber, K.-P., Berthold, M.R.: Analysis of Simulation Models with Fuzzy Graph Based Metamodeling. Performance Evaluation 27/28, 473–490 (1996)

8. Ferrer-Troyano, F., Aguilar-Ruiz, J.S., Riquelme, J.C.: Incremental Rule Learning based on Example Nearness from Numerical Data Streams. In: Proc. of the 20th Annual ACM Symp. on Applied Computing, Santa Fe, NM, USA, pp. 568–572. ACM Press, New York (2005)
9. Blake, C., Merz, C.: UCI Repository of Machine Learning Databases (compiled in 1998), ftp://ftp.ics. uci.edu/pub/machine-learning-databases
10. Metz, C.E.: Basic principles of ROC Analysis. Seminars in Nuclear Medicine 8, 283–298 (1978)

A Partial Correlation-Based Algorithm for Causal Structure Discovery with Continuous Variables

Jean-Philippe Pellet[1,2] and André Elisseeff[1]

[1] IBM Research
Business Optimization Group
Säumerstr. 4, 8803 Rüschlikon, Switzerland
{jep,ael}@zurich.ibm.com
[2] Swiss Federal Institute of Technology Zurich
Machine Learning Group
Institute of Computational Science
Universitätstr. 6, 8092 Zurich, Switzerland

Abstract. We present an algorithm for causal structure discovery suited in the presence of continuous variables. We test a version based on partial correlation that is able to recover the structure of a recursive linear equations model and compare it to the well-known PC algorithm on large networks. PC is generally outperformed in run time and number of structural errors.

1 Introduction

Detecting causation from observational data alone has long been a controversial issue. It is not before the pioneering work of Pearl and Verma [1] and Spirtes et al. [2] that causal discovery was formalized theoretically and linked with a graphical representation: directed acyclic graphs (DAGs).

PC,[1] the reference causal discovery algorithm, is based on conditional independence (CI) tests. While such a test can be implemented efficiently with discrete variables, it is not generalizable to the continuous case straightforwardly. With the assumption that variables are jointly distributed according to a multivariate Gaussian, we know that a test for zero partial correlation[2] is equivalent to a CI test [3]. In this paper, we present an algorithm based on partial correlation that is faster and makes fewer errors than PC on datasets with more than a few hundred samples.

In section 2, we review principles of causal discovery, pose the problem, and mention related work. We then present the algorithm in section 3 and analyze its complexity. Experimental results are shown and discussed in section 4. We conclude in section 5 and include proofs and definitions in appendix A.

[1] "PC" stands for "Peter" and "Clark" after the inventors of the method [2].
[2] Definition in appendix A.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 229–239, 2007.

2 Background and Problem Statement

The search for the true causal structure underlying some data set is of paramount importance when the effect actions rather than predictions are to be returned. By focusing on predictions only, a system cannot address problems where some parts of the data distribution process is changed. Causality analysis is a mean to address these nonstationary problems by computing the mechanism generating the data and by assessing the effect of some changes in that mechanism. The first step to a causal analysis is the definition of the causal structure represented as a DAG. In general, this problem is impossible to solve with observational data only. Causal structures can be retrieved only up to some equivalence class: besides (undirected) adjacencies, only colliders, i.e., triples of variables where one is a common effet of two causes, can be specified exactly.

There are mainly two classes of causal discovery algorithms: score-based and constrained-based. In this paper, we are concerned with the second type only. PC is a typical constraint-based algorithm. We present its high-level description, also known as the IC algorithm [1]:

1. For each variable pair (X, Y) in the set of variables $\mathbf{V}$, look for a set $\mathbf{S}_{XY}$ such that X and Y are conditionally independent given $\mathbf{S}_{XY}$: $(X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY})$;[3] add an edge between X and Y if no such set can be found;
2. For each pair (X, Y) with a common neighbor Z, turn the triple into a V-structure $X \rightarrow Z \leftarrow Y$ if $Z \in \mathbf{S}_{XY}$;
3. Propagate the arrow orientation to preserve acyclicity without introducing new V-structures.

Because of the subset search in Step 1, PC and IC have an exponential time complexity in the worst case.

Current causal discovery algorithms assume that the underlying causal structure is a Bayesian network (with discrete variables), i.e., that the dataset is *DAG-isomorphic* [4]. Extensions have been developed for the case of continuous variables when the underlying causal structure is a linear structural equation model (SEM). In SEMs, we describe each variable $x_i = f_i(\mathbf{pa}_i, u_i)$ as a function of its parents and a random disturbance term. When the corresponding graph is acyclic, the SEM is said to be *recursive*.

In this paper, we focus on SEMs and on continuous variables, frequent in econometrics, social sciences, and health care, for instance. Focusing on SEMs rather than Bayesian networks avoids the computational difficulties of handling continuous conditional probability distributions. But one of the main challenges to solve is to find a convincing statistical test of CI for continuous variables. To simplify our task, we solve the simpler case of a linear recursive SEM, where each functional equation is of the form $x_i = \langle \mathbf{w}_i, \mathbf{pa}_i \rangle + u_i$. Imposing a Gaussian distribution on the disturbance terms u_i yields a multivariate Gaussian distribution over $\mathbf{V}$, and a partial correlation $\rho_{XY \cdot \mathbf{Z}}$ will be zero if and only if $(X \perp\!\!\!\perp Y \mid \mathbf{Z})$

[3] We have $(X \perp\!\!\!\perp Y \mid \mathbf{Z}) \iff P(X = x \mid \mathbf{Z} = \mathbf{z}) = P(X = x \mid Y = y, \mathbf{Z} = \mathbf{z})$.

holds. Thus, testing for zero partial correlation is a valid conditional independence test for continuous variables in a linear recursive SEM with uncorrelated Gaussian disturbance terms.

Related Work. Partial correlation has been used extensively in econometrics and social sciences in path analysis with relatively small models (e.g., [5]). In causal discovery, it has only been used (as transformed by Fisher's z, see [6,7]) as a continuous replacement for CI tests designed for discrete variables and assuming a small conditioning set size.

Causal graph construction, especially if considered as determination of the Markov blanket of each variable, can be assimilated to a feature selection task for each node. Other causal algorithms performing a search to retrieve the Markov blanket of single variables include MMMB [8] and HITON_MB [9]. These papers also discuss the link to feature selection. But to the best of our knowledge, none of them has been extended and applied to fully-continuous datasets.

Other approaches to learning the structure of causal or Bayesian networks with continuous variables without first discretizing them include using a CI test from Margaritis [10]. This test, however, is very computationally expensive, which limits its use even for medium-sized problems. Work has also been done in score-based approaches for learning with continuous [11] and mixed [12] variables and integrating expert knowledge in the form of priors, but they do not provide a theoretical proof that the obtained graph is a perfect map of the dataset.

3 Total Conditioning for Causal Discovery

Whereas PC removes edges from a full graph as CI is found, our Total Conditioning (TC) method starts with an empty graph and adds edges between two nodes when conditioning on all the others does not break any causal dependency.

1. For each pair (X, Y), add an edge $X - Y$ if the partial correlation $\rho_{XY \cdot \mathbf{V} \setminus \{X,Y\}}$ does not vanish. We obtain the moral graph of $\mathcal{G}_0$, i.e., an undirected copy of $\mathcal{G}_0$ where all parents of the colliders are pairwise linked;
2. Remove spurious links between parents of colliders introduced in Step 1 and identify V-structures;
3. Propagate constraints to obtain maximally oriented graph (completed PDAG).

Partial correlations in Step 1 can be computed efficiently by inverting the correlation matrix $\mathbf{R}$. With $\mathbf{R}^{-1} = (r^{ij})$, we have: $\rho_{X_i X_j \cdot \mathbf{V} \setminus \{X_i, X_j\}} = -r^{ij} / \sqrt{r^{ii} r^{jj}}$.

In terms of Gaussian Markov random fields (a special case of undirected graphical models), Step 1 constructs the correct graph by adding edges where the total partial correlation is significantly different from zero (see, e.g., [13]). In the case of the DAG-isomorphic problems we handle, Step 1 builds the correct structure up to moral graph equivalence: it will actually build the correct undirected links and marry all parents. This means that every original V-structure will be turned

into a triangle pattern. Step 3 is common to several algorithms constructing the network structure under CI constraints [1,2]. Step 2 is a local search looking for orientation possibilities. To explain it, we need the following definition.

Definition 1. *In an undirected graph* $\mathcal{G} = (\mathbf{V}, \mathbf{E})$*, let* $\mathbf{Tri}(X - Y)$ *(with* $X, Y \in \mathbf{V}$ *and* $(X, Y) \in \mathbf{E}$*) be the set of vertices forming a triangle with* X *and* Y*. Suppose that* $\mathcal{G}$ *is the moral graph of the DAG representing the causal structure of some DAG-isomorphic dataset. A set of vertices* $\mathbf{Z} \subset \mathbf{Tri}(X - Y)$ *then has the* Collider Set *property for the pair* (X, Y) *if it is the largest set that fulfills*

$$\exists \mathbf{S}_{XY} \subset \mathbf{V} \setminus \{X, Y\} \setminus \mathbf{Z} : (X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY}) \tag{1}$$

$$\textit{and} \quad \forall Z_i \in \mathbf{Z} : (X \not\perp\!\!\!\perp Y \mid \mathbf{S}_{XY} \cup Z_i)\,. \tag{2}$$

Step 2 looks at each edge that is part of some triangle and determines if it is spurious due to a V-structure effect. This is exactly the case when two variables X, Y in a triangle X, Y, Z can be made conditionally independent by a set that does not contain Z. A search is then performed for each of those edges to determine a set $\mathbf{Z} \subset \mathbf{Tri}(X - Y)$ that has the Collider Set property, using a small search space for $\mathbf{S}_{XY}$ and $\mathbf{Z}$ as allowed by the result of Step 1. If this search is successful, the edge $X - Y$ is removed and the detected V-structures properly oriented for each collider. Practically, the search for $\mathbf{S}_{XY}$ can be restricted to a subset of the union of the Markov blankets for X and Y, and the search for $\mathbf{Z}$ is restricted by definition to $\mathbf{Tri}(X - Y)$, which make both tasks tractable, unless the graph has a high connectedness.

Algorithm 1. The Total Conditioning algorithm

Input: $D : p \times n$ dataset with p n-dimensional data points
Output: $\mathcal{G}$: maximally oriented partially directed acyclic graph

$\mathcal{G} \leftarrow$ empty graph with n nodes
for each unordered pair X, Y **do**
  if $\rho_{XY \cdot \mathbf{V} \setminus \{X,Y\}}$ does not vanish **then** add link $Y - X$ to $\mathcal{G}$
end for
for each edge $X - Y$ part of a fully-connected triangle **do**
  if $\exists \mathbf{Z} \subset \mathbf{Tri}(X - Y)$ that satisfies the Collider Set property **then**
    remove link $X - Y$ from $\mathcal{G}$
    for each $Z_i \in \mathbf{Z}$ **do** orient edges as $X \rightarrow Z_i \leftarrow Y$
  end if
end for
perform constraint propagation on $\mathcal{G}$ to obtain completed PDAG
return $\mathcal{G}$

Complexity Analysis Step 1 has a complexity of $\mathcal{O}(n^3)$, which comes from the matrix inversion needed to compute the partial correlations. Step 2 has a complexity $\mathcal{O}(n^2 2^\alpha)$, where $\alpha = \max_{X,Y} |\mathbf{Tri}(X - Y)| - 1$. Step 3 is $\mathcal{O}(n^3)$. The

overall complexity is then $\mathcal{O}(n^3 + n^2 2^\alpha)$, depending on the value of α as determined by the structure of the graph to be recovered. In the worst case of a fully-connected graph, it is, like PC, exponential in the number of variables.

After removal of the spurious links and the usual constraint propagation [1,2], the returned graph is the maximally-oriented PDAG of the Markov equivalence class of the generating DAG $\mathcal{G}_0$.

In the appendix, we prove the correctness of TC; i.e., we show that in the large-sample limit and with reliable statistical tests, TC converges to the actual perfect map of the dataset to be analyzed, up to its equivalence class.

Significance Tests. A particularly delicate point in this algorithm is the statistical test deciding whether a partial correlation is significantly different from zero. In a network of n nodes, Step 1 performs $n(n-1)/2$ tests for determining the undirected skeleton. On average, we will then falsely reject the null hypothesis $\rho = 0$ about $\alpha n(n-1)/2$ times, and thus include as many wrong edges in the graph. We then set the significance level for the individual tests to be inversely proportional to $n(n-1)/2$ to avoid this problem, without noticing an increase in the Type II error rate experimentally. The PC algorithm does not suffer from this issue because of the detailed way of repeatedly testing for edge existence with increasing conditioning set cardinality.

In practice, we replaced the more traditional Fisher approximate z-transform of the sample correlation by t-tests on the beta weights of the corresponding linear regression equations, whose distributions are known to be Gaussian with zero mean under the null hypothesis $\rho = 0$ (see, e.g., [14], p. 243).

4 Experimental Results

The performance of the TC algorithm was evaluated against the PC algorithm [2] where CI tests were replaced by zero partial correlation tests. We were unable to compare it to newer algorithms like SCA [15] or MMHC [16] because generalizing them to handle continuous variables require techniques that are too computationally expensive. We used the following networks (from the Bayes net repository):

- Alarm, 37 nodes, 46 arcs, 4 undirected in the PDAG of the equivalence class. It was originally designed to help interpret monitoring data to alert anesthesiologists to various situations in the operating room.
- Hailfinder, 56 nodes, 66 arcs, 17 undirected in its PDAG. It is a normative system that forecasts severe summer hail in northeastern Colorado.
- A subset of Diabetes, with 104 nodes, 149 arcs, 8 undirected in its PDAG, which was designed as a preliminary model for insulin dose adjustment.

The graphs were used as a structure for a linear SEM. The parentless variables were sampled as Gaussians with zero mean and unit standard deviation; the other variables were defined as a linear combination of their parents with coefficient randomly distributed uniformly between 0.1 and 0.9, similarly to what

was done in [6]. The disturbance terms were also normally distributed. We used the implementation of PC proposed in the BNT Structure Learning Matlab package [17], where we set the statistical significance of the tests to $\alpha = 0.05$. The implementation of TC was also done in Matlab; all experiments were run on a 2 GHz machine.

Fig. 1 (a) shows the training errors for PC and TC against the number of samples for Alarm. For each sample size, 9 datasets were drawn from the model; the error bars picture the standard deviation over these 9 runs. Starting at about 150 samples, TC outperforms PC. It introduces at most one unnecessary arc and misses between 0 and 3. On average, TC was about 20 times faster than the implementation of PC we used, although the factor tended to decrease with larger sample sizes; see Fig. 1 (b).

The results for Hailfinder are shown in Fig. 2. The results for PC are sparser than for TC, because of its long run times. In order to speed it up, we set the maximum node fan-in parameter to 6, so that PC would not attempt to conduct CI tests with conditioning sets larger than 6. For large datasets, we could run PC only once, so that we have little information on the variance of its results for this network, but even if we average the five last PC results for between 550 and 10000 samples, TC does better for each of its runs on this range. PC still beats TC on sample sizes smaller than 200. We also see on Fig. 2 (b) how the fan-in parameter imposed an upper bound on the run times of PC.

Fig. 3 shows errors and run times for Diabetes (note that the sample size starts from 200, because in the case where we have fewer samples than the number of variables, TC would have to inverse a matrix that does not have full rank). Again, PC does better at first, and starting at 500 samples, it is outperformed in accuracy. The difference of the number of errors stabilizes around 5 or 6. Our run times are still significantly shorter.

Finally, Fig. 4 shows the results of an experiment where we took the first n nodes of Diabetes for a fixed sample size of 1000 in order to show the response of the algorithms to an increasing number of variables in networks of similar structure. Results show that PC makes 1 to 2 mistakes fewer on the smaller networks but is outperformed for $n > 50$ on this particular instance. Although the run times of PC are still significantly higher, rescaling the plots shows that the increase of n increases the run times of both algorithms by a very similar factor for all tested graph sizes.

Discussion. TC consistently beats PC when the sample size gets larger, and does so in a small fraction of the time needed by PC. In particular, PC is slowed down by nodes with a high degree, whereas TC handles them without the exponential time complexity growth if they are not part of triangles, as in Hailfinder. In general, TC resolves all CI relations (up to married parents) in $\mathcal{O}(n^3)$ in Step 1, whereas all PC can do in $\mathcal{O}(n^3)$ is resolve CI relations with conditioning sets of cardinality 1. It is then reasonable to expect TC to scale better than PC on sparse networks where nodes have a small number of parents.

PC could not be beaten on small sample sizes. It is yet an unsolved challenge for TC to handle problems where the number of variables exceeds the number of

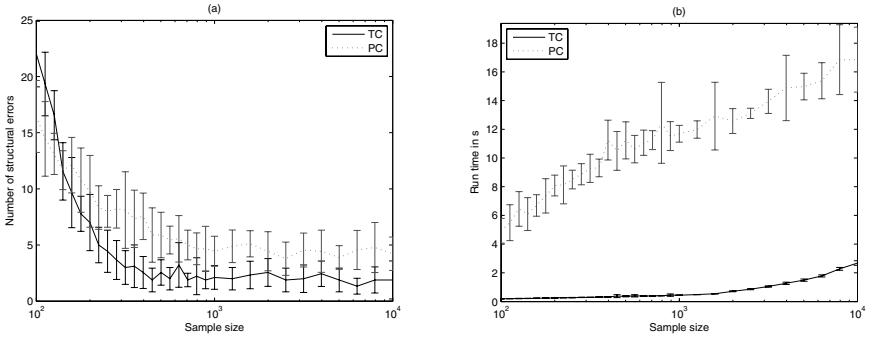


Fig. 1. Alarm: (a) structural errors and (b) run times as a function of sample size

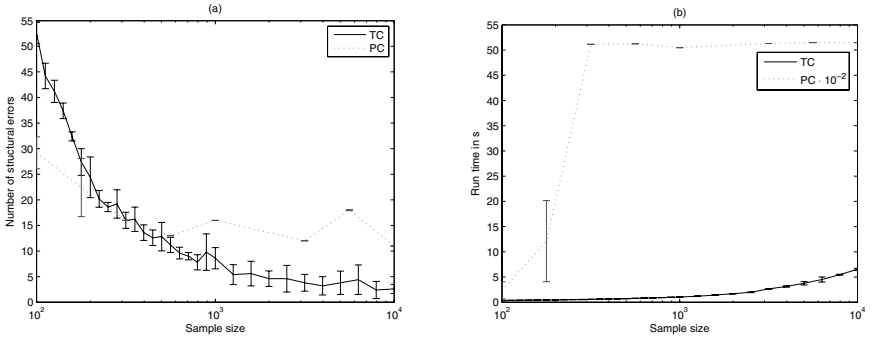


Fig. 2. Hailfinder: (a) structural errors and (b) run times as a function of sample size

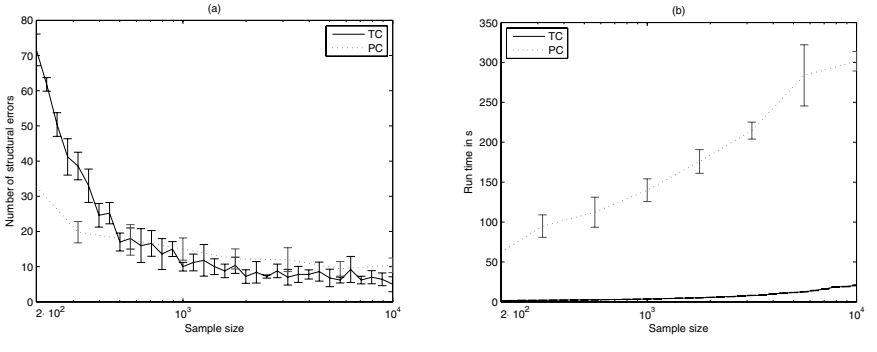


Fig. 3. Diabetes: (a) structural errors and (b) run times as a function of sample size

samples, as in gene expression networks, thus leading to an attempt at inverting a matrix that does not have full rank. Regularizing the covariance matrix might help make TC more robust in this case. PC and TC are complementary in the

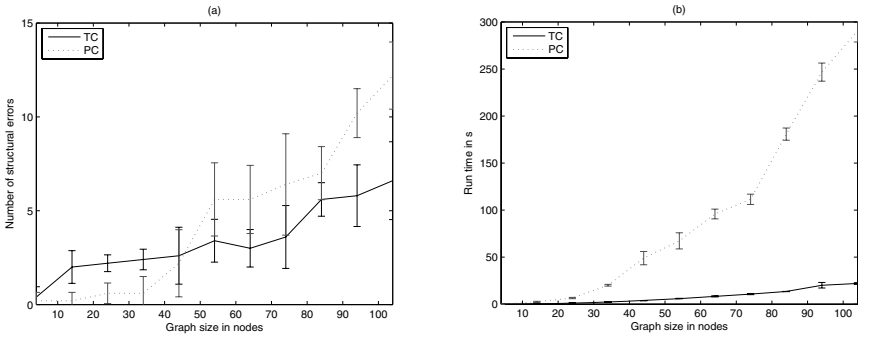


Fig. 4. Diabetes: (a) structural errors and (b) run times as a function of n

sense that PC is preferably used with smaller sample sizes, and TC can take over more accurately with larger datasets.

5 Conclusion

Causal discovery with continuous variables is tractable with the multivariate Gaussian assumption and partial correlation: we showed an algorithm based on it to recover the exact structure in the large sample limit. The algorithm first checks for each pair of variables if their association can be accounted for by the intermediate of other variables, and if not, links them, thus determining the Markov blanket of each node. A second pass performs a local search to detect the V-structure and orient the graph correctly.

The proposed algorithm outperforms or equals the reference PC algorithm in accuracy (except for very small sample sizes) in a fraction of its run time. In the future, we intend to investigate further the behavior of the algorithm and improve it in these conditions. We will also work on generalizing partial correlation and the underlying linear regression to the nonlinear case.

References

1. Pearl, J., Verma, T.: A theory of inferred causation. In: Proc. of the Second Int. Conf. on Principles of Knowledge Representation and Reasoning, Morgan Kaufmann, San Francisco (1991)
2. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search, vol. 81. Springer Verlag, Berlin (1993)
3. Baba, K., Shibata, R., Sibuya, M.: Partial correlation and conditional correlation as measures of conditional independence. Australian & New Zealand Journal of Statistics 46(4) (2004)
4. Wong, S.K.M., Wu, D., Lin, T.: A structural characterization of dag-isomorphic dependency models. In: Proc. of the 15th Conf. of the Canadian Society for Computational Studies of Intelligence, pp. 195–209. Morgan Kaufmann, San Francisco (2002)

5. Alwin, D.F., Hauser, R.M.: The decomposition of effects in path analysis. American Sociological Review 40(1) (1975)
6. Scheines, R., Spirtes, P., Glymour, C., Meek, C., Richardson, T.: The tetrad project: Constraint based aids to causal model specification. Technical report, Carnegie Mellon University, Dpt. of Philosophy (1995)
7. Schäfer, J., Strimmer, K.: Learning large-scale graphical gaussian models from genomic data. In: AIP Conference Proceedings, vol. 776, pp. 263–276 (2005)
8. Tsamardinos, I., Aliferis, C.F., Statnikov, A.: Time and sample efficient discovery of markov blankets and direct causal relations. In: Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, ACM Press, New York (2003)
9. Aliferis, C.F., Tsamardinos, I., Statnikov, A.: Hiton, a novel markov blanket algorithm for optimal variable selection. In: Proceedings of the 2003 American Medical Informatics Association (AMIA) Annual Symposium, pp. 21–25 (2003)
10. Margaritis, D.: Distribution-free learning of bayesian network structure in continuous domains. In: Proc. of the 20th National Conf. on AI (2005)
11. Geiger, D., Heckerman, D.: Learning gaussian networks. Technical Report MSR-TR-94-10, Microsoft Research (1994)
12. Bøttcher, S.: Learning bayesian networks with mixed variables. In: Proceedings of the Eighth International Workshop in Artificial Intelligence and Statistics (2001)
13. Talih, M.: Markov Random Fields on Time-Varying Graphs, with an Application to Portfolio Selection. PhD thesis, Hunter College (2003)
14. Judge, G.G., Hill, R.C., Griffiths, W.E., Lütkepohl, H., Lee, T.C.: Introduction to the Theory and Practice of Econometrics, 2nd edn. Wiley, Chichester (1988)
15. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using bayesian networks to analyze expression data. In: RECOMB, pp. 127–135 (2000)
16. Tsamardinos, I., Brown, L.E., Aliferis, C.F.: The max-min hill-climbing bayesian network structure learning algorithm. Machine Learning (2006)
17. Leray, P., François, O.: Bnt structure learning package (2004)

A Appendix: Correctness Proof

For all proofs, we assume the given dataset D is DAG-isomorphic.

Definition 2. Partial correlation *between variables X and Y given a set of variables $\mathbf{Z}$ is the correlation of the residuals R_X and R_Y resulting from the linear regression of X on $\mathbf{Z}$ and of Y on $\mathbf{Z}$, respectively.*

Definition 3. *In an DAG $\mathcal{G}$, two nodes X, Y are* d-separated *by $\mathbf{Z} \subset \mathbf{V} \setminus \{X, Y\}$, written $(X \nleftrightarrow Y \mid \mathbf{Z})$, if every path from X to Y is blocked by $\mathbf{Z}$. A path is* blocked *if at least one diverging or serially connected node in in $\mathbf{Z}$ or if at least one converging node and all its descendants are not in $\mathbf{Z}$.*

If X and Y are not d-separated *by $\mathbf{Z}$, they are* d-connected: *$(X \leftrightarrow Y \mid \mathbf{Z})$. This is generalized to sets $\mathbf{X}, \mathbf{Y}$: $(\mathbf{X} \nleftrightarrow \mathbf{Y} \mid \mathbf{Z})$ holds if pairwise separation holds for all i, j: $(X_i \nleftrightarrow Y_j \mid \mathbf{Z})$.*

Lemma 1. *In a DAG $\mathcal{G}$, any (undirected) path π of length $\ell(\pi) > 2$ can be blocked by conditioning on any two consecutive nodes in π.*

Proof. It follows from Def. 3 that a path π is blocked when either at least one collider (or one of its descendants) is not in $\mathbf{S}$, or when at least one non-collider is in $\mathbf{S}$. It therefore suffices to show that conditioning on two consecutive nodes always includes a non-collider. This is the case because two consecutive colliders would require bidirected arrows, which is a structural impossibility with simple DAGs. □

Lemma 2. *In a DAG $\mathcal{G}$, two nodes X, Y are* d*-connected given all other nodes $\mathbf{S} = \mathbf{V} \setminus \{X, Y\}$ if and only if any of the following conditions holds:*

(i) There is an arc from X to Y or from Y to X (i.e., $X \to Y$ or $X \leftarrow Y$);
(ii) X and Y have a common child Z (i.e., $X \to Z \leftarrow Y$).

Proof. We prove this by first proving an implication and then its converse.

($\Longleftarrow$) If (i) holds, then X and Y cannot be d-separated by any set. If (ii) holds, then Z is included in the conditioning set and d-connects X and Y by Def. 3.

($\Longrightarrow$) X and Y are d-connected given a certain conditioning set when at least one path remains open. Using the conditioning set $\mathbf{S}$, paths of length > 2 are blocked by Lemma 1 since $\mathbf{S}$ contains all nodes on those paths. Paths of length 2 contain a mediating variable Z between X and Z; by Def. 3, $\mathbf{S}$ blocks them unless Z is a common child of X and Y. Paths of length 1 cannot be blocked by any conditioning set. So the two possible cases where X and Y will be d-connected are (i) or (ii). □

Corollary 1. *Two variables X, Y are dependent given all other variables $\mathbf{S} = \mathbf{V} \setminus \{X, Y\}$ if and only if any of the following conditions holds:*

(i) X causes Y or Y causes X;
(ii) X and Y have a common effect Z.

Proof. It follows directly from Lemma 2 due to the DAG-isomorphic structure, which ensures that there exists a DAG where CI and d-separation map one-to-one. Lemma 2 can then be reread in terms of CI and causation instead of d-separation and arcs. □

Lemma 3. *The subset $\mathbf{Z}$ that has the Collider Set property for the pair (X, Y) is the set of all direct common effects of X and Y and exists if and only if X is neither a direct cause nor a direct effect of Y.*

Proof. The fact that $\mathbf{Z}$ exists if and only if X is neither a direct cause nor a direct effect of Y is a direct consequence of (1), which states that X and Y can be made conditionally independent. This is in contradiction with direct causation. We now assuming that some $\mathbf{S}_{XY}$ and $\mathbf{Z}$ have been found.

($\Longrightarrow$) By (1) and (2), we know that each Z_i opens a dependence path between X and Y (which are independent given $\mathbf{S}_{XY}$) by conditioning on $\mathbf{S}_{XY} \cup Z_i$. By Def. 3, conditioning on Z_i opens a path if Z_i is either a colliding node or one of its descendants. As, by definition, $\mathbf{Z} \subset \mathbf{Tri}(X - Y)$, we are in the first case. We conclude that Z_i is a direct effect of both X and Y.

($\Longleftarrow$) Note that (1) and (2) together are implied in presence of a V-structure $X \rightarrow Z_i \leftarrow Y$. Thus, a direct effect is compatible with the conditions. The fact that $\mathbf{Z}$ captures all direct effects follows from the maximization of its cardinality. □

Theorem 1. *If the variables are jointly distributed according to a multivariate Gaussian, TC returns the PDAG of the Markov equivalence class of the DAG representing the causal structure of the data-generating process.*

Proof. An edge is added in Step 1 between X and Y if we find that $\rho_{XY \cdot \mathbf{V} \setminus \{X,Y\}} \neq 0$. We conclude $(X \not\perp\!\!\!\perp Y \mid \mathbf{V} \setminus \{X, Y\})$, owing to the multivariate Gaussian distribution. Corollary 1 says that this implies that X causes Y or Y causes X, or that they share a common child. Therefore, each V-structure is turned into a triangle by Step 1. Step 2 then examines each link $X - Y$ part of a triangle, and by Lemma 3, we know that if the search for a set $\mathbf{Z}$ that has the Collider Set property succeeds, there must be no link between X and Y. We know by the same lemma that this set includes all colliders for the pair (X, Y), so that all V-structures are correctly identified. Step 3 is the same as in the IC or PC algorithms; see Pearl and Verma [1,2]. □

Visualizing Sets of Partial Rankings

Antti Ukkonen

Helsinki University of Technology,
Laboratory of Computer and Information Science, HIIT BRU

Abstract. Partial rankings are totally ordered subsets of a set of items. They arise in different applications, such as clickstream analysis and collaborative filtering, but can be difficult to analyze with traditional data analysis techniques as they are combinatorial structures. We propose a method for creating scatterplots of sets of partial rankings by first representing them in a high-dimensional space and then applying known dimensionality reduction methods. We compare different approaches by using quantitative measures and demonstrate the methods on real data sets from different application domains. Despite their simplicity the proposed methods can produce useful visualizations that are easy to interpret.

1 Introduction

A *partial ranking* is a totally ordered, typically smallish subset of a larger set of items. For example, the large set might contain titles of all movies that came out in 2006, while a partial ranking contains only movies seen by one individual, ranked best to worst according to the individual's opinions of the movies. A collection of partial rankings would thus contain the preferences of a number of people, who have all seen a different subset of the entire set of movies.

Partial rankings can be found in a number of applications, such as clickstream analysis, collaborative filtering, certain voting systems and different scientific applications. Good data visualization techniques can provide important insight to the phenomenon described by the data. The human visual system is often much more efficient in discovering dependencies and structure from visual representations than most automated data analysis systems, provided the visualization is accurate and truthful.

The visualization task we address in this work is the following: given a collection of partial rankings, create a two dimensional scatterplot where each partial ranking is represented by a single point and similar rankings are close to each other. These visualizations can be useful for gaining understanding of the structure of the data, such as making assessments of the number and shape of clusters.

An example of this is given in Figure 1. Here the set of items is the set of integers $1, 2, \ldots, 12$, and each partial ranking contains 6 items. The rankings in column C_1 tend to have a 7 or a 9 in the first position and a 6 or a 4 in the last. On the other hand, the rankings in column C_2 have typically a 2 or a 5 in the first position and mostly end with a 1 or an 11. Clearly the rankings can be divided to two clusters. The scatterplot in Figure 1 shows two well separated

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 240–251, 2007.

C_1	C_2
8 9 10 5 12 6	5 10 4 12 1 11
9 10 5 3 11 6	5 2 3 4 12 11
8 9 3 5 11 6	6 3 8 4 11 7
9 8 5 11 12 2	5 2 4 8 1 9
7 9 10 5 3 6	2 12 8 7 9 1
7 1 5 2 6 4	2 5 4 3 7 1
1 9 3 5 2 12	5 2 6 4 12 1
9 5 2 11 12 4	2 6 10 4 9 1
7 8 3 12 2 4	5 10 3 9 7 11
9 8 3 5 2 4	5 6 10 12 9 7
9 10 3 12 2 11	5 3 8 9 11 1
7 10 2 12 11 6	5 10 3 4 1 11
7 6 12 2 11 4	2 6 3 8 9 1
1 8 11 12 6 4	2 6 8 12 7 11
7 9 8 5 6 4	5 10 12 7 1 11

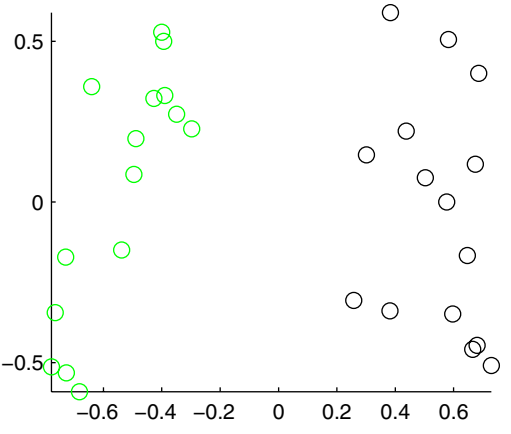


Fig. 1. Example visualization of the partial rankings displayed on the left. The scatterplot is created using PCA and the hypersphere representation (Sect. 2.2). The rankings are generated by a model with two components (C_1 and C_2). In the visualization rankings originating from the same component are indicated by the same color.

groups of circles, which correspond to the partial rankings in columns C_1 and C_2. Thus, in this simple example we can correctly identify the cluster structure of the set of partial rankings from the visualization.

For instance, if the partial rankings reflect user preferences the visualization might reveal if the users can be segmented to groups based on the rankings. Also, suppose that in addition to the rankings some demographic information is known about the users. Then it is easy to see whether certain demographics tend to have homogenous preferences. In case of the movie example one might conjecture that older people prefer (to some extent) different movies than teenagers, for example. In the visualization this would mean that points corresponding to rankings given by adults are in a separate region from the points corresponding to rankings given by teenagers. Furthermore, if we obtain a new ranking without the auxiliary information, we might be able to infer some of its properties by considering the neighboring rankings (the properties of which are known) in the visualization.

To visualize partial rankings, we first map them to a high-dimensional space and then use existing dimension reduction methods to obtain a two dimensional representation of the rankings. Numerous algorithms for dimension reduction have been proposed, we use PCA (see any book on multivariate statistics), Local MDS [8], and Isomap [6], as they together cover many features typically found in dimension reduction algorithms.

We also present a generating model for partial rankings that assumes some fixed number of underlying components that each generate partial rankings independent of the other components. This way we can take two approaches to evaluate the quality of a visualization. On one hand we can measure how well partial rankings generated by different components can be separated from those

generated by other components in the visualization. On the other hand we can consider only the local neighborhood of a partial ranking in the high dimensional space and see how this is preserved by the projection.

Algorithms for dimension reduction require as input either the high dimensional vectors or a function that defines interpoint distances in the high dimensional space. If the feature vectors are available, we can compute the interpoint distances easily by using some suitable metric. We present two different representations of partial rankings as high dimensional vectors. The first representation is motivated by the problem that arises when we want to measure the distance between two partial rankings that have no items in common. The second one has a geometric interpretation as the partial rankings are mapped to the surface of a hypersphere.

The rest of this paper is organized as follows. Section 2 discusses two different ways of representing partial rankings in a high-dimensional space. The generating model is described in Section 3 and our measures for evaluating the quality of a visualization are defined in Section 4. Experimental results are given in Section 5 while Section 6 is a short conclusion.

2 Representations of Partial Rankings

In order to apply a dimension reduction method on the set D of partial rankings, we must devise a suitable representation for the data. These representations are also used in [7], where the goal is quite different from the this paper: [7] looks for clusters high likelihood and does not consider aspects related to visualization at all. We discuss necessary definitions and notation at first.

Let M be a finite set of *items* to be ranked, and let n denote the size of M. A *partial ranking* ϕ is a totally ordered subset of M. An item u is *covered* by ϕ if it belongs to the subset of M that ϕ orders. We write $u \in \phi$ if the item u is covered by ϕ. Denote by l the *length* of ϕ, that is, the number of items that it covers. We let $\phi(u)$ denote the *rank* of item u if ϕ covers u, otherwise $\phi(u)$ is not defined. The *reverse* of ϕ, denoted ϕ^R, is a partial ranking that covers the same items as ϕ but orders them in exactly the opposite way. Furthermore, a partial ranking is a partial order[1] on the subset of M it covers. This means we can view ϕ as a set of ordered pairs. The pair (u, v) belongs to ϕ, denoted $(u, v) \in \phi$, whenever $\phi(u) < \phi(v)$, that is, ϕ ranks u before v.

2.1 Graph Representation

The *agreement graph* of D, denoted $G_a(D)$, is an undirected graph with the partial rankings as vertices. An edge connects the partial rankings ϕ and ψ, if and only if $(u, v) \in \phi \cap \psi$ for some u and v and there is no u' and v' such that $(u', v') \in \phi$ and $(v', u') \in \psi$. The *disagreement graph* of D, denoted $G_d(D)$, is also an undirected graph with the partial rankings as vertices. An edge connects

[1] A reflexive, transitive and asymmetric binary relation.

the partial rankings ϕ and ψ, if and only if $(u, v) \in \phi$ and $(v, u) \in \psi$ for some u and v. Intuitively, two partial rankings are connected in $G_a(D)$ if they agree at least on the order of one pair of items and have no disagreeing pairs. Likewise, two partial rankings are connected in $G_d(D)$ if the partial rankings disagree on the order of at least one pair of items. They are allowed to have agreeing pairs, however.

Denote the adjacency matrices of the agreement and disagreement graphs by $G_a(D)$ and $G_d(D)$ as well. Consider the matrix $X = G_a(D) - G_d(D)$. We define the graph representation of ϕ as the row of X that corresponds to ϕ, denoted $X(\phi, :)$. As the similarity of two fragments is reflected in the number of common neighbors they have in $G_a(D)$ and $G_d(D)$, the vectors $X(\phi, :)$ and $X(\psi, :)$ tend to be positively correlated when ϕ and ψ are similar, and negatively correlated when ϕ and ψ are dissimilar.

The motivation for this representation is based on the assumption that a partial ranking is actually a part of a complete ranking (one that covers all of M) and that the partial rankings in D have been generated by several complete rankings. Two partial rankings can be generated by the same complete ranking (and hence should be considered in some sense similar) but still have no items in common.

For example, let $M = \{a, b, c, d, e, f, g, h\}$, and suppose $\phi = a < b < c < d$ and $\psi = e < f < g < h$ have been generated by the same permutation. Establishing that this is the case based on ϕ and ψ alone is not possible, but suppose D contains also the partial rankings $c < d < e < f$ and $a < b < g < h$. Both ϕ and ψ are connected to these in $G_a(D)$. As the number of common neighbors of ϕ and ψ increases in both $G_a(D)$ and $G_d(D)$, the likelier it becomes that they indeed are generated by the same permutation and should therefore be considered similar.

Note that the dimensionality of the graph representation grows linearly in the size of D and constructing the agreement and disagreement graphs takes time $O(|D|^2)$.

2.2 Hypersphere Representation

As the graph representation does not scale too well with increasing size of D, we consider next an another approach that behaves better in this respect. The basic idea is to map the partial rankings to points on the surface of an n-dimensional hypersphere. Thus the dimensionality of the representation is limited by $|M|$ instead of $|D|$. Moreover, it turns out that computing this representation from a given set of partial rankings scales linearly in the size of D.

Let f be a mapping from D to $\mathbb{R}^n$ and let d be a distance measure in $\mathbb{R}^n$. Intuitively, the distance between ϕ and ϕ^R should be the maximum distance between any two partial rankings. Hence, we want to have (1) $d(f(\phi), f(\phi^R)) = \max_\psi\{d(f(\phi), f(\psi))\}$. Moreover, all partial rankings should be treated equally in this respect, which means we must have (2) $d(f(\phi), f(\phi^R)) = d(f(\psi), f(\psi^R))$ for all ϕ and ψ as well. Finally, $d(f(\phi), f(\phi)) = 0$ for all ϕ.

It is not hard to see that the above requirements are satisfied when f maps the partial rankings to points on the surface of a hypersphere. For the *complete rankings* π this is easily accomplished by using a method similar to the Borda score (see e.g. [5]). We represent the partial ranking ϕ as an n-dimensional vector, one dimension for each item in M, so that the value at the dimension corresponding to u depends on the rank $\phi(u)$. Let $f_\pi = f(\pi)$ and denoting by $f_\pi(u)$ the coordinate that corresponds to item u in the n-dimensional vector representation we want to construct, define

$$f_\pi(u) = Z(-\frac{n-1}{2} + \pi(u) - 1), \tag{1}$$

where Z is a normalization constant so that the length of f_π equals 1. Requirements (1) and (2) are satisfied when d is for example the cosine distance (one minus the cosine of the angle between two vectors).

With partial rankings the situation is conceptually more complicated, as it is not obvious what value to assign those coordinates that correspond to items $u \notin \phi$. The solution turns out to be very easy, however.

We can view the partial ranking ϕ as a *partial order* that ranks those items that belong to ϕ and leaves the mutual order between all remaining items unspecified. A *linear extension* of the partial order ϕ is a total order that ranks u before v whenever $(u, v) \in \phi$. There are in total $n!/l!$ different linear extensions of ϕ, when ϕ is a partial ranking of length l. As the linear extensions are complete rankings, we can use Equation (1) to find their representations in $\mathbb{R}^n$.

We define the representation $f_\phi = f(\phi)$ of ϕ as the *center of these points*, i.e., $f(\phi)$ is the mean of all $f(\pi)$ normalized to unit length, where π is a linear extension of ϕ. It can be shown that this is obtained when we let

$$f_\phi(u) = Q(-\frac{l-1}{2} + \phi(u) - 1) \tag{2}$$

for all $u \in \phi$, and otherwise let $f_\phi(u) = 0$. Here Q is a normalization constant and l is the length of ϕ. Computing this can be done in time $O(l|D|)$.

3 A Generative Model for Partial Rankings

In this section we briefly discuss a generating model for partial rankings. This model is used in the experiments to evaluate the quality of the visualizations by measuring how well certain properties of the model are preserved by the projection, and how the quality of a visualization is affected by different parameters of the model.

We assume the partial rankings are generated by a population of individuals. This population can be segmented to groups $j = 1, \ldots, k$ so that members of a certain group j have similar preferences. This is modeled by a group specific partial order B_j on M. Suppose an individual i from group j had access to all of M (i.e., had seen all movies that came out in 2006). Then i can in theory specify a total order π on the items according to his preferences. As i belongs to

group j, we assume π is a linear extension of B_j. However, since in general an individual can only evaluate a subset of the items (those movies he has seen), i can specify only the partial ranking ϕ that covers the subset known to him, but ranks those as π would.

The generating model works as follows: initialize the model by picking k partial orders B_j on M. Then for each individual, first pick one B_j, then pick one linear extension π of B_j, and finally pick a subset of l items and create the partial ranking by projecting π on this subset. In each case we use a uniform distribution on the respective sample space.

To simplify matters computationally, we select the B_js from a restricted class of partial orders that are called either *rankings with ties* [2] or *bucket orders* [3]. These are essentially totally ordered, disjoint subsets ("buckets") of the complete set of items. For example, we could say that in a given group all five star movies are in the first subset, all four star movies in the second and so on, with the one star movies in the last subset. A parameter of the model is thus also the number of buckets in a bucket order, denoted b.

4 The Quality of a Visualization

Evaluating the quality of a visualization can be done both qualitatively and quantitatively. In this work we concentrate on using quantitative measures. Our approach is to view visualization as an *information retrieval* task, as recently proposed in [9]. This allows us to assess the visualization in terms of *precision* and *recall*, which are typically used to evaluate IR systems. Precision is defined as the fraction of relevant documents in the set of retrieved documents. Recall is the fraction of relevant documents retrieved.

To use this framework for evaluating a visualization we must define what points constitute the set of relevant and retrieved partial rankings given some query. We assume that the data analysis task to be solved by using the visualization calls for the identification of similar partial rankings to a given query ranking. Hence the set of relevant rankings can be defined either as i) those that have been generated by the same component of the generating model or ii) those that are close to the query ranking in the high dimensional space. Making this distinction allows us to separately measure how well the visualizations preserve global and local features. Finally, the set of retrieved rankings are those that are close to the query ranking in the visualization.

In the experiments we will measure precision in terms of the components of the generating model, denoted $p_c(i)$ and the local neighborhoods of the high dimensional representations, denoted $p_n(i)$. Let $O_q(i)$ be the set of q closest points to point i in the high dimensional space. Likewise, let $P_w(i)$ be the set of w closest points to i in the visualization. Finally, denote by $C(i)$ the set of partial rankings that have been generated by the same component as i. We have thus

$$p_c(i) = \frac{|C(i) \cap P_w(i)|}{|P_w(i)|} \quad \text{and} \quad p_n(i) = \frac{|O_q(i) \cap P_w(i)|}{|P_q(i)|}. \tag{3}$$

The final precision values for a given visualization are averages of $p_c(i)$ and $p_n(i)$ over all possible i.

We use a different distance function depending on the representation to find the set of relevant rankings. With the hypersphere representation we use cosine distance, and with the graph representation we use correlation. Distances between points in the visualization is always measured using the Euclidean distance.

5 Experiments

The algorithms we compare in the experiments are PCA, Local MDS [8], and Isomap [6]. Local MDS is run with $\lambda = 0$ (in this case Local MDS corresponds to Curvilinear Component Analysis [1]) and $\lambda = 0.5$. These algorithms were chosen as they are well established, and as they together cover most of the typical approaches found in dimensionality reduction methods, such as linear projection (PCA), nonlinear projection by minimizing the differences in distances (Local MDS) and manifold embedding (Isomap).

5.1 Artificial Data

We use artificial data to examine how varying the parameters of the generating model affects the quality of the visualizations. Parameters of interest are the length of a partial ranking l, the number of components in the model (k) and the number of buckets per component (b). The total number of partial rankings was 500 in each case. The neighborhood size w required for computing the precision values was set to 50. This is because we assume that when people inspect the visualizations they tend to look at the general surroundings of a "query" point and not just the few closest neighbors. The set of relevant rankings for computing p_n is $O_{20}(i)$, that is, we view the 20 closest rankings as being relevant.

Results for the hypersphere and graph representations are shown in figures 2 and 3, respectively. The plotted values are averages of 20 rounds. A new input was generated on every round. Both of the representations behave very similarly. This observation is not obvious, as the graph and hypersphere representations are based on rather different approaches. In general using the hypersphere representation seems to give slightly better results.

In both figures it is immediately seen that the curves behave as expected as the parameters of the generating model are varied. With small values of l and b and high values of k the data is inherently more difficult to analyze as there is less information available. This is reflected by a worse performance in the visualizations. Especially the differences can be seen in case of the component-wise precision p_c.

Another clear phenomenon is that Local MDS with $\lambda = 0$ (represented by a plus sign) outperforms all other methods by a fair margin when it comes to p_n. Conversely, the same method turns out to be the poorest choice when p_c is considered. PCA gives constantly best performance with p_c. This difference

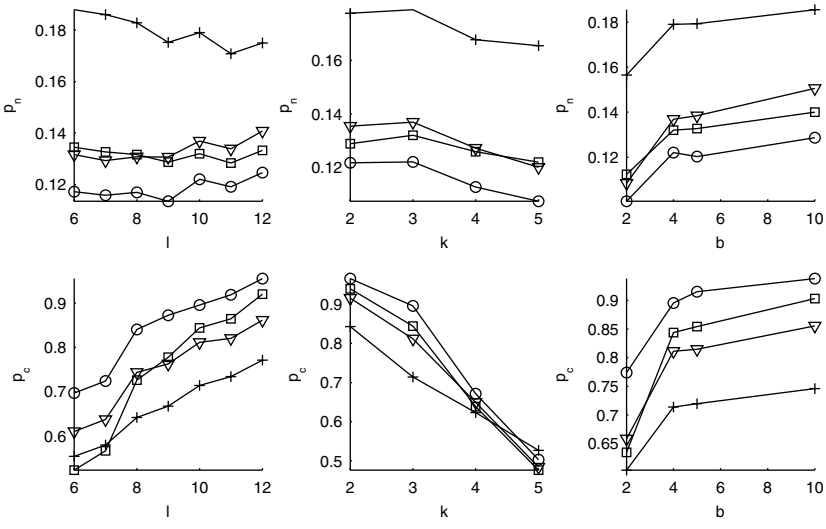


Fig. 2. Characteristics of the visualizations based on the hypersphere representation using different algorithms for dimension reduction. PCA: circle, IM: square, LMDS $\lambda = 0$: plus and LMDS $\lambda = 0.5$: triangle. l: length of a partial ranking, k number of components in the input, b number of buckets per component.

is easily explained by considering how the methods operate: PCA attempts to preserve a global structure by creating a maximum variance projection whereas LMDS, especially with $\lambda = 0$, explicitely tries to preserve the local neighborhood of a point. Increasing λ to 0.5 improves the performance of LMDS slightly in terms of p_c but dramatically decreases it in terms of p_n.

5.2 Voting, Clickstream and Preference Data

We consider three real world data sets as examples. The first one is voting data from the 2002 general election in Ireland. The data is collected in the electoral district of northern Dublin and is publicly available[2]. Each voter is allowed to rank as many candidates in order of preference. The data consists of these rankings. The total number of candidates is 12. We selected a subset of the entire data where each voter had ranked at least 4 and at most 6 candidates. The average number of candidates in a vote is 4.7.

The second data contains rankings of 17 categories of the MSNBC portal[3]. Each ranking corresponds to the sequence in which a user browsed through different areas of the portal. If the user visited the same category several times we only consider the first occurrence of a category. The data was pruned to retain only partial rankings that contain at least 6 and at most 8 different categories. The average number of categories visited by a user is 6.5.

[2] http://www.dublincountyreturningofficer.com/
[3] http://kdd.ics.uci.edu/databases/msnbc/

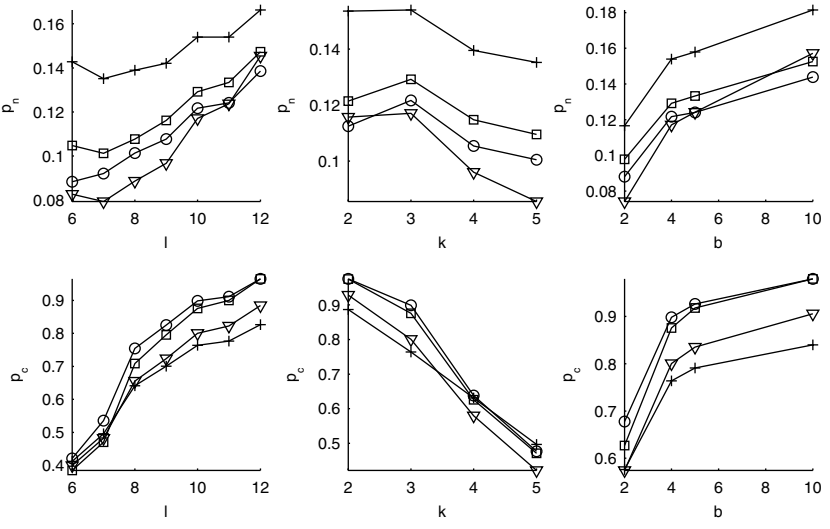


Fig. 3. Characteristics of the visualizations based on the graph representation using different algorithms for dimension reduction. PCA: circle, IM: square, LMDS $\lambda = 0$: plus and LMDS $\lambda = 0.5$: triangle. l: length of a partial ranking, k number of components in the input, b number of buckets per component.

In both cases we computed a clustering of the partial rankings by using the method presented in [7]. The number of clusters found in the voting and MSNBC data sets is 2 and 3, respectively. We use the clusterings obtained this way when evaluating the component specific precision p_c. These results may vary slightly as the clustering algorithm can produce different results on separate runs.

Figure 4 shows example visualizations containing a random subset of 500 rankings of both data sets using different dimension reduction techniques. Partial rankings were represented using the hypersphere method. Graphs on the right side indicate how p_c and p_n behave when the neighborhood size in the visualization is increased. The LMDS based methods are superior with both p_c and p_n. This could be due to the probable absence of truly distinct clusters in the data sets. Hence both p_c and p_n measure essentially the preservation of the local neighborhood. For large neighborhoods the methods tend to produce equally good results.

Our third data set[4] is from a survey that studied people's preferences towards different types of sushi [4]. The complete data contains 5000 partial rankings of length 10, one for each participant. In addition to the rankings, also demographic information, such as gender, age and place of residence is included for each respondent. A subset of this data is visualized using LMDS with $\lambda = 0$ in Figure 5 as an example. The subset contains rankings given by respondents who were both born and are currently living in the eastern part of Japan. Moreover,

[4] `http://www.kamishima.net/sushi/`

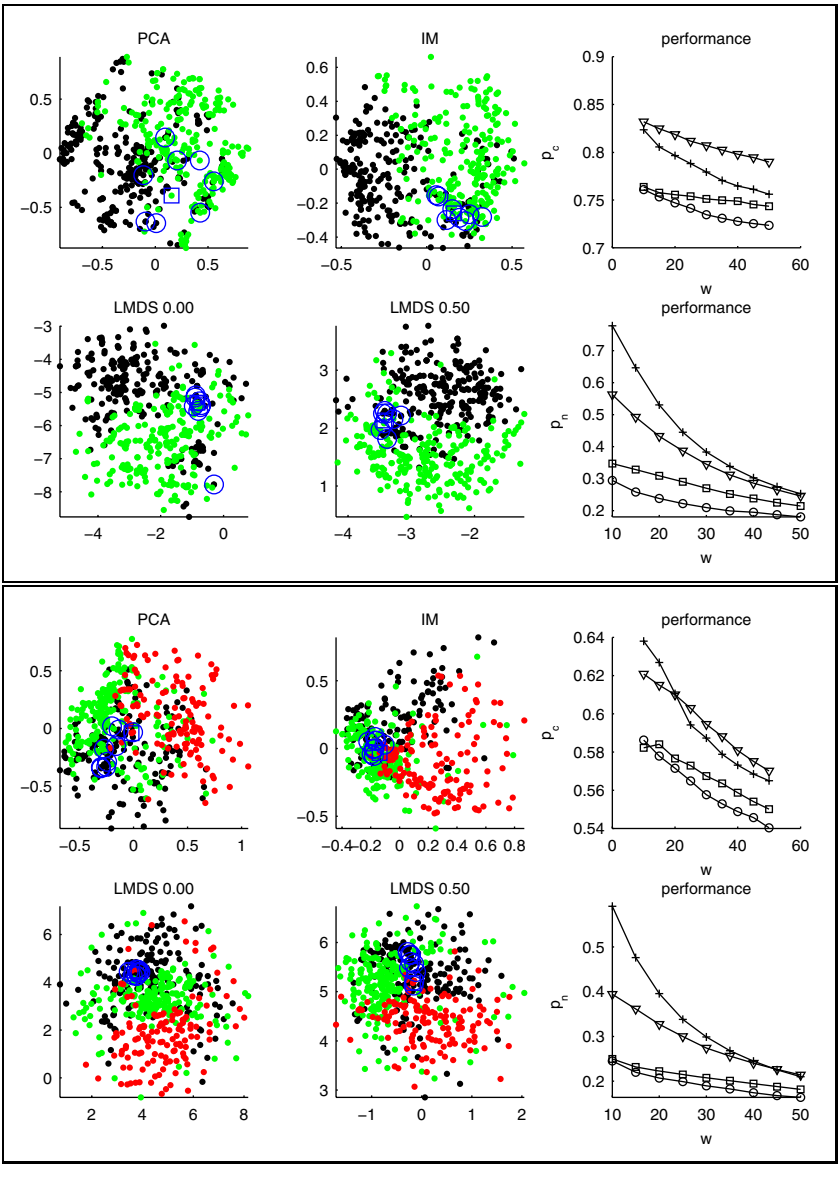


Fig. 4. Example visualizations of a voting data (above) and a clickstream data (below). Each dot in the scatterplots corresponds to one partial ranking. Similar rankings are plotted next to each other. The voting data is assumed to have two clusters while the clickstream data has three. In both cases ten closest neighbors (in the input space) of the point indicated by a surrounding square are indicated by circles. This gives an intuition how the local neighborhood of a point is preserved in the projection. For the graphs on the right PCA: circle, IM: square, LMDS $\lambda = 0$: plus and LMDS $\lambda = 0.5$: triangle.

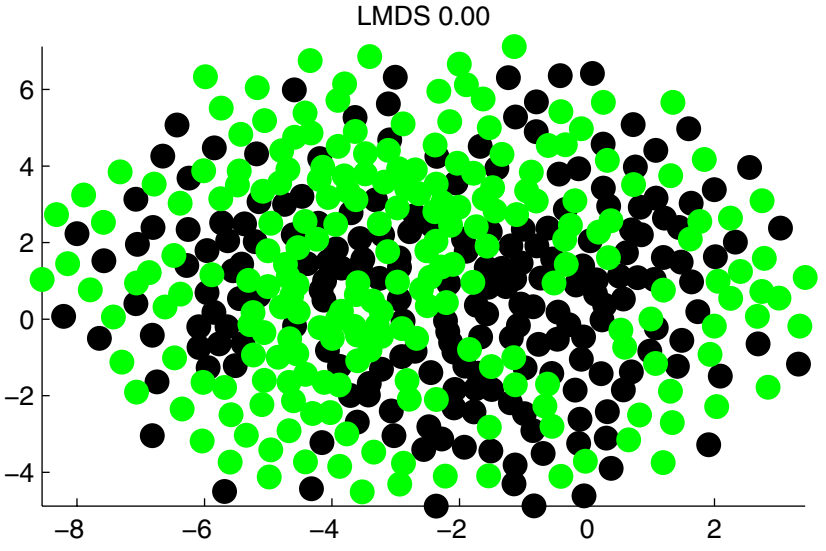


Fig. 5. A visualization of a subset of a sushi preference data. Rankings of respondents who indicated themselves as being 15-19 years old are plotted in light gray (or green) while the black dots correspond to respondents who reported themselves as being 50 years or older.

we included only teenagers (aged 15-19 years, indicated in light gray/green) and people aged over 50 (indicated in black) to the subset. Even though the groups are globally overlapping, we can easily identify local areas that tend to contain mostly respondents belonging to one of the groups.

6 Conclusions

We have considered the problem of creating two dimensional visualizations of a set of partial rankings. The approach was to first represent the partial rankings in a high dimensional space and then apply a dimension reduction method to obtain the visualization. We showed that this approach leads to potentially useful visualizations, the quality of which depends both on the characteristics of the input data and the method used. Using PCA for dimension reduction tends to preserve global features better whereas using MDS based methods preserve local features. Perhaps somewhat surprisingly both high dimensional representations for partial rankings that were discussed result in very similar visualizations, even though the representations are based on very different principles. In general, sets of rankings that contain less information about the order of the items are more difficult to visualize.

References

1. Demartines, P., Herault, J.: Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. IEEE Transactions on Neural Networks 8(1), 148–154 (1997)
2. Fagin, R., Kumar, R., Mahdian, M., Sivakumar, D., Vee, E.: Comparing and aggregating rankings with ties. In: Proceedings of the 23rd ACM Symposium on Principles of Database Systems (PODS), ACM Press, New York (2004)
3. Gionis, A., Mannila, H., Puolamaki, K., Ukkonen, A.: Algorithms for discovering bucket orders from data. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data mining, pp. 561–566. ACM Press, New York (2006)
4. Kamishima, T., Akaho, S.: Efficient Clustering for Orders. In: Proceedings of The 2nd International Workshop on Mining Complex Data, pp. 274–278 (2006)
5. Moulin, H.: Axioms of Cooperative Decision Making. Cambridge University Press, Cambridge (1991)
6. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
7. Ukkonen, A., Mannila, H.: Finding representative sets of bucket orders from partial rankings. review (submitted)
8. Venna, J., Kaski, S.: Local multidimensional scaling. Neural Networks 19(6–7), 889–899 (2006)
9. Venna, J., Kaski, S.: Nonlinear dimensionality reduction as information retrieval. In: Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (2007)

A Partially Supervised Metric Multidimensional Scaling Algorithm for Textual Data Visualization

Ángela Blanco and Manuel Martín-Merino*

Universidad Pontificia de Salamanca
C/Compañía 5, 37002, Salamanca, Spain
ablancogo@upsa.es, mmartinmac@upsa.es

Abstract. Multidimensional Scaling Algorithms (MDS) allow us to visualize high dimensional object relationships in an intuitive way. An interesting application of the MDS algorithms is the visualization of the semantic relations among documents or terms in textual databases.

However, the MDS algorithms proposed in the literature exhibit a low discriminant power. The unsupervised nature of the algorithms and the 'curse of dimensionality' favor the overlapping among different topics in the map. This problem can be overcome considering that many textual collections provide frequently a categorization for a small subset of documents.

In this paper we define new semi-supervised measures that reflect better the semantic classes of the textual collection considering the a priori categorization of a subset of documents. Next the dissimilarities are incorporated into the Torgerson MDS algorithm to improve the separation among topics in the map. The experimental results show that the model proposed outperforms well known unsupervised alternatives.

1 Introduction

Visualization algorithms are multivariate data analysis techniques that help to discover the underlying structure of high dimensional data. Several techniques have been proposed to this aim such as Correspondence Analysis [15], Self Organizing Maps (SOM) [17,13] or Multidimensional Scaling algorithms (MDS) [10]. In particular, the Torgerson MDS algorithm has been successfully applied to real problems because it is based on an efficient and robust linear algebra operation such as the Singular Value Decomposition (SVD) [10].

An interesting application of the Torgerson MDS algorithm is the visualization of the semantic relations among terms or documents [15,18] in text mining problems. This visual representation gives more information than just the hard classification of terms or documents and is particularly helpful for novel users [9]. However, the word maps generated by most MDS algorithms have a low discriminant power. Thus, due to the unsupervised nature of the algorithms and to the 'curse of dimensionality' the specific terms tend to overlap strongly in the center map disregarding their semantic meaning [6,19]. Then, the relations induced by the map become often meaningless.

* Corresponding author.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 252–262, 2007.

The visualization of term relationships has been usually done in the literature by non-supervised techniques [15]. Moreover, common semi-supervised algorithms [14] can not be applied because the categorization of a small subset of terms is a complex and time consuming task [1,20]. However, textual databases provide often a classification for a subset of documents [1] because this is easier for human experts. Therefore, the organization of terms into topics can only be improved considering a small subset of labels in the space of documents.

In this paper we propose a semi-supervised version of the Torgerson MDS algorithm that improves the visualization of the term relationships taking advantage of the available labels in the space of documents. To this aim, we first introduce a new class of semi-supervised measures that take into account a categorization for a subset of documents via the Mutual Information [24]. Next, the Torgerson MDS algorithm is applied to generate the word map considering this similarity matrix. Notice that common semi-supervised clustering and visualization algorithms proposed in the literature [21,7,5] take into account the class labels modifying the error function to be optimized. This results frequently in complex non-linear optimization problems. Our approach introduces partial supervision via the definition of a semi-supervised similarity and does not modify the error function. Therefore, the semi-supervised algorithm proposed keeps the nice properties of the original Torgerson MDS algorithm.

This paper is organized as follows. In section 2 the Torgerson MDS algorithm is introduced. Section 3 presents the new semi-supervised MDS algorithm. In section 4 the algorithm is applied to the visualization of term relationships. Finally section 5 gets conclusions and outlines future research trends.

2 The Torgerson MDS Algorithm

Let $\boldsymbol{X}(n \times d)$ be a matrix of n objects represented in $\mathbb{R}^d$ and $\boldsymbol{D} = (\delta_{ij})$ the dissimilarity matrix made up of the object proximities. The Multidimensional Scaling algorithms look for an object configuration in a low dimensional space (usually two for visualization) in such a way that the inter-pattern Euclidean distances reflect approximately the original dissimilarity matrix.

A large variety of algorithms have been proposed in the literature. In this paper we have considered the Torgerson MDS algorithm [10] because it exhibits several properties interesting for text mining problems. First, the algorithm with the Euclidean distance is equivalent to a linear PCA [10] that can be solved efficiently through a Singular Value Decomposition (SVD) [4]. Second, the optimization problem doesn't have local minima. Notice that many MDS algorithms such as Sammon or certain neural based techniques [17] rely on non-linear optimization methods that can get stuck in local minima. Finally, the Torgerson MDS algorithm can be considered with certain similarities equivalent to the Latent Semantic Indexing (LSI) [3] that has been successfully applied in text mining problems.

Next we introduce briefly the Torgerson MDS algorithm. For a detailed explanation see [10].

Define the matrix $\boldsymbol{A}(n \times n)$ as $[\boldsymbol{A}]_{ij} = a_{ij} = -\frac{1}{2}\delta_{ij}^2$. The inner product matrix $\boldsymbol{B}$ can be obtained as:

$$\boldsymbol{B} = \boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{H}\boldsymbol{A}\boldsymbol{H} \tag{1}$$

where $\boldsymbol{H}$ is a centering matrix defined as:

$$\boldsymbol{H} = \boldsymbol{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T \tag{2}$$

with $\mathbf{1} = [1\,1\ldots1]^T$ a column matrix of n ones and $I(n \times n)$ the identity matrix. The Torgerson MDS algorithm looks for a projection $W : \mathbb{R}^d \rightarrow \mathbb{R}^k$ to a lower dimensional space such that the Euclidean distances in $\mathbb{R}^k$ preserve as much as possible the original dissimilarities. The object coordinates that verify this condition are given [10] by:

$$\boldsymbol{X}_k = \boldsymbol{V}_k \boldsymbol{\Lambda}_k^{\frac{1}{2}}\,, \tag{3}$$

where $\boldsymbol{V}_k$ is the $n \times k$ orthonormal matrix whose columns are the kth first eigen vectors of $\boldsymbol{B}$ and $\boldsymbol{\Lambda}_k = diag(\lambda_1, \ldots, \lambda_k)$ is a diagonal matrix with λ_i the ith eigenvalue of $\boldsymbol{B}$. The object coordinates in equation (3) can be obtained through a SVD. This operation is particularly efficient when only the first eigenvectors are needed as it happens for visualization purposes.

3 A Semi-supervised MDS Algorithm

The word maps generated by the Torgerson MDS algorithm often suffer from a low discriminant power. The unsupervised nature of the algorithm favors the overlapping between different topics in the map. Moreover, due to the "curse of dimensionality" the words concentrate around the center map and the smaller distances become often meaningless [19,6].

In this section we explain how the categorization of a subset of documents by human experts can be exploited to improve the word maps generated by the MDS algorithm. The novelty of this problem relies in that we are trying to improve an unsupervised technique that works in the space of terms considering the available labels in the space of documents. To this aim rather than modifying the error function as is usually done by supervised clustering and visualization algorithms [21,7,5] we define a semi-supervised similarity that takes into account the class labels. This similarity will reflect both, the semantic classes of the textual collection and the term relationships inside each class. Once the semi-supervised dissimilarities are computed, the Torgerson MDS algorithm can be applied to generate a visual representation of the term relationships. Notice that our approach allow us to extend to the semi-supervised case any algorithm that works from a dissimilarity matrix.

Let t_i, t_j be two terms and $\{C_k\}_{k=1}^c$ the set of document categories created by human experts. The association between terms and categories of documents

is usually evaluated in the Information Retrieval literature by the Mutual Information [24] defined as:

$$I(t_i; C_k) = \log \frac{p(t_i, C_k)}{p(t_i)p(C_k)}\,, \tag{4}$$

where $p(t_i, C_k)$ denotes the joint cooccurrence probability of term t_i and class C_k. $p(t_i)$, $p(C_k)$ are the a priori probability of occurrence of term t_i and class C_k respectively. The Mutual Information is able to capture non-linear relationships between terms and categories.

However, it has been pointed out in the literature [24] that the index (4) gives higher score to rare terms. To overcome this problem we have considered a weighted version of the previous index defined as

$$I'(t_i; C_k) = p(t_i, C_k) \log \frac{p(t_i, C_k)}{p(t_i)p(C_k)}\,. \tag{5}$$

This index reduces obviously the weight of the less frequent terms.

Now, we can define a similarity measure between terms considering the document class labels. This measure will be referred to as supervised similarity from now on. Obviously, this similarity should become large for terms that are related/unrelated with the same categories of documents. This suggests the following definition for the term similarity:

$$s_1(t_i, t_j) = \frac{\sum_k I'(t_i; C_k) I'(t_j; C_k)}{\sqrt{\sum_k (I'(t_i; C_k))^2}\sqrt{\sum_k (I'(t_j; C_k))^2}}\,. \tag{6}$$

The numerator of this similarity will become large for terms that are correlated with similar categories. Notice that the index (6) can be considered a cosine similarity between the vectors $I'(t_i; \cdot) = [I'(t_i; C_1), \ldots, I'(t_i; C_c)]$ and $I'(t_j; \cdot) = [I'(t_j; C_1), \ldots, I'(t_j; C_c)]$. This allow us to interpret the new similarity as a non-linear transformation to a feature space [23] where a cosine similarity is computed. Other dissimilarities can be considered in feature space but we have chosen the cosine because it has been widely used in the Information Retrieval literature [16]. Finally the similarity (6) is translated and scaled so that it takes values in the interval $[0, 1]$.

The similarity defined above can be considered an average over all the categories. Next, we provide an alternative definition for the supervised similarity that considers only the class with higher score. It can be written as

$$s_2(t_i, t_j) = \max_k \{\bar{I}(t_i; C_k) * \bar{I}(t_j; C_k)\}\,, \tag{7}$$

where $\bar{I}$ is a normalized Mutual Information defined as

$$\bar{I}(t_i; C_k) = \frac{I(t_i; C_k)}{\max_l \{I(t_i; C_l)\}}\,. \tag{8}$$

This normalization factor guarantees that $s_2(t_i, t_i) = 1$. The similarity (7) will get large when both terms are strongly correlated with one of the classes.

The supervised measures proposed earlier will score high terms that are related with the same categories. However, for visualization purposes it is also interesting to reflect the semantic relations among the terms inside each class or among the main topics. This information is provided by unsupervised measures such as for instance the cosine. This justifies the definition of a semi-supervised similarity as a convex combination of a supervised and an unsupervised measure. This similarity will reflect both, the semantic groups of the textual collection and the term relationships inside each topic. It is defined as follows:

$$s(t_i, t_j) = \lambda s_{sup}(t_i, t_j) + (1 - \lambda) s_{unsup}(t_i, t_j)\,, \tag{9}$$

where s_{sup} and s_{unsup} denote the supervised and unsupervised measures respectively. The parameter λ verifies $0 \leq \lambda \leq 1$. This parameter will determine if the resulting map reflects better the semantic classes of the textual collection (λ large) or the semantic relations among the terms (λ small).

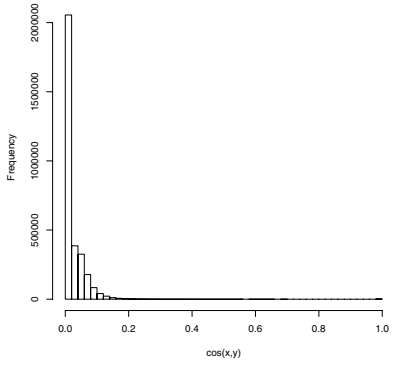


Fig. 1. Cosine similarity histogram

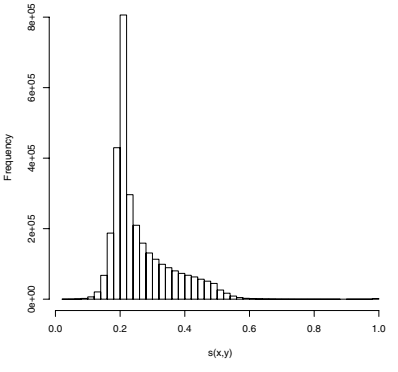


Fig. 2. Histogram of the average semi-supervised similarity measure

The semi-supervised similarity (9) has an interesting property that is worth to mention. Figure 1 shows the similarity histogram for an unsupervised measure such as the cosine while figure 2 shows the histogram for a semi-supervised one. The unsupervised measure (fig. 1) is strongly degraded by the 'curse of dimensionality'. The histogram is very skew and most of the similarities are zero or close to zero [2]. Thus, MDS algorithms will put together terms in the map just because they are far away from the same subset of words. Hence, small distances in the map could suggest false term relationships. Additionally, the map will show a spherical distribution that has nothing to do with the underlying structure of the data [6]. On the other hand the standard deviation for the semi-supervised similarity (fig. 2) is larger than for the cosine similarity and the histogram is smoother. This suggests that the semi-supervised similarity is more robust to

the 'curse of dimensionality' and consequently any algorithm based on distances will perform better [19,6].

Finally the Torgerson MDS algorithm introduced in section 2 is applied to derive a visual representation of the semi-supervised similarities. To this aim, the similarity (9) must be transformed into a dissimilarity using for instance the following rule $\delta_{ij} = 1 - s_{ij}$ [10]. Figure 2 suggests that for the textual collection considered in this paper the semi-supervised measure defined gives rise to an inner product matrix $\boldsymbol{B}$ semi-definite positive. Therefore the Torgerson MDS algorithm can be used to get an approximate representation of the data in a space of dimension $< n - 1$ where n is the sample size.

The semi-supervised MDS algorithm presented earlier assumes that the whole textual collection is categorized by human experts. However, it is very common in text mining problems that only a small percentage of the textual collection is labeled [1]. Hence, we have a small training set of categorized documents and a much larger test set of documents not labeled. A large variety of techniques have been proposed in the literature to work this kind of datasets. In this paper we have considered the Transductive Support Vector Machines (TSVM) [23] because they have been successfully applied to the categorization of document collections [11]. The Transductive SVM aims at finding a decision function that maximizes the margin of both, labeled and unlabeled patterns. This technique allow us to reduce significantly the misclassification error of the inductive SVM, particularly when the training set is very small [11]. Once the documents are classified using the TSVM, the semi-supervised similarity (9) is computed as in the supervised case. However, those terms that appear in less than five documents categorized by human experts or by the TSVM are considered unreliable and the similarity is computed in an unsupervised way ($\lambda = 0$).

4 Experimental Results

In this section we apply the proposed algorithms to the construction of word maps that visualize term semantic relationships. The textual collection considered, is made up of 2000 *scientific abstracts* retrieved from three commercial databases 'LISA', 'INSPEC' and 'Sociological Abstracts'. For each database a thesaurus created by human experts is available. Therefore, the thesaurus induces a classification of terms according to their semantic meaning. This will allow us to exhaustively check the term associations created by the map.

Assessing the performance of algorithms that generate word maps is not an easy task. In this paper the maps are evaluated from different viewpoints through several objective functions. This methodology guaranty the objectivity and validity of the experimental results.

The objective measures considered in this paper quantify the agreement between the semantic word classes induced by the map and the thesaurus. Therefore, once the objects have been mapped, they are grouped into topics with a clustering algorithm (for instance PAM [12]). Next we check if words assigned

to the same cluster in the map are related in the thesaurus. To this aim we have considered the following objective measures:

- F measure [16]: It is a compromise between 'Recall' and 'Precision' and it has been widely used by the Information Retrieval community. Intuitively, F measures if words associated by the thesaurus are clustered together in the map.
- Entropy measure [22]: It measures the uncertainty for the classification of words that belong to the same cluster. Small values suggest little overlapping between different topics in the maps. Therefore smaller values are considered better.
- Mutual Information [22]: It is a non-linear correlation measure between the word classification induced by the thesaurus and the word classification given by the clustering algorithm. Notice that this measure gives more weight to specific terms [24] and therefore provides a valuable information about changes in the position of less frequent terms.

Table 1. Empirical evaluation of several semi-supervised visualization algorithms for a collection of scientific abstracts

	F	E	I
Torgerson MDS	0.46	0.55	0.17
Least square MDS	0.53	0.52	0.16
Torgerson MDS (Average)	0.69	0.43	0.27
Torgerson MDS (Maximum)	0.77	0.36	0.31
Least square MDS (Average)	0.70	0.42	0.27
Least square MDS (Maximum)	0.76	0.38	0.31

Table 1 shows the experimental results for the semi-supervised MDS algorithms proposed in this paper. Two unsupervised techniques have been considered as reference, the Torgerson MDS algorithm introduced in section 2 and a standard least square MDS algorithm [10]. For each technique, two semi-supervised similarities have been considered, the average (see equation (6)) and the maximum (see equation (7)). As unsupervised measure we have selected the cosine because it has been widely used by the information retrieval community [16] with reasonable good results.

The least square MDS has been always initialized by a PCA to avoid that the algorithm get stuck in a local minima. The λ parameter in the semi-supervised measures has been set up to 0.5 which achieves a good balance between structure preservation and topic separation in the map. Increasing the value of λ will improve the separation among different topics in the map. However, this will distort the relationships induced for terms related to the same topic. Conversely, smaller values of λ will favor the overlapping among topics moving toward the unsupervised case. Therefore, λ is a user defined parameter that depends on the

problem at hand. From the analysis of table 1 the following conclusions can be drawn:

- The semi-supervised techniques improve significantly the word maps generated by the unsupervised ones. In particular, the semi-supervised Torgerson MDS (rows 3-4) reduces significantly the overlapping among the different topics in the map (E is significantly reduced). The Mutual Information is particularly improved which suggests that the overlapping among the specific terms that belong to different topics is reduced in the map. Finally, the F measure corroborates the superiority of the proposed algorithm.

  The least square MDS algorithm (rows 5-6) improves similarly the maps generated when the semi-supervised dissimilarities are considered. This suggests that many algorithms that work from a dissimilarity measure can benefit from the ideas presented in this paper.
- The maximum semi-supervised similarity gives always better results than the average. This can be explained because the maximum supervised similarity is defined considering only the class that is more correlated with the terms. This feature improves the separation of the topics in the map.

Finally figure 3 illustrates the performance of the semi-supervised Torgerson MDS algorithm from a qualitative point of view. For the sake of clarity only a

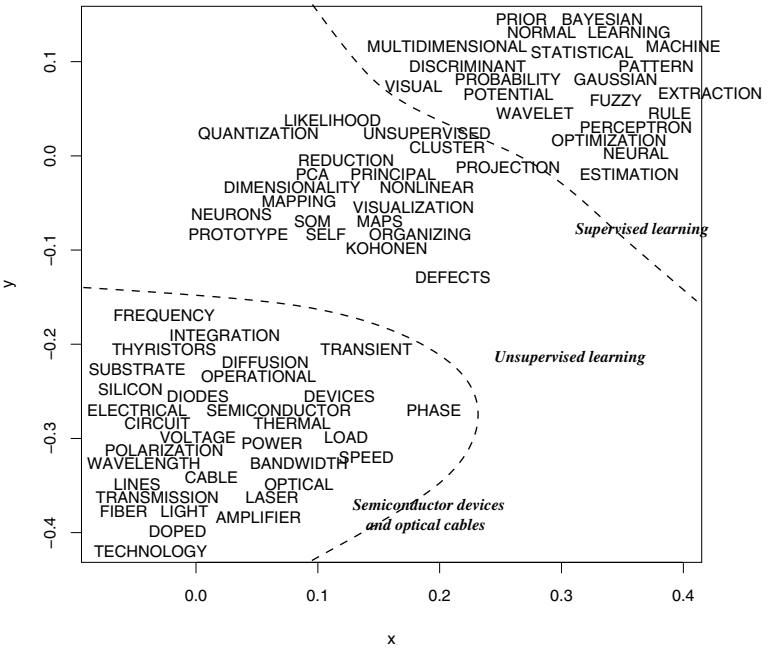


Fig. 3. Map generated for the semi-supervised Torgerson MDS algorithm with average similarity for a subset of words that belong to three topics with different overlapping

subset of words that belong to three topics have been drawn. We report that the term associations induced by the map are satisfactory and that the semantic topics can be easily identified in the map.

As we have mentioned earlier, in text mining applications only a small subset of documents is categorized by human experts. Therefore, from a practical point of view it is very important to evaluate the sensibility of the method proposed to the percentage of categorized documents. The Transductive SVM has been implemented using the SVM$^{\text{light}}$ software. For the multicategory classification we have considered the 'one against one' approach. The regularization parameters C and C^* for training and test sets respectively have been set up to one. Finally, $\lambda = 0.5$ in the semi-supervised measures. The empirical results suggest that this value allow us to identify easily the semantic topics in the word maps and the term relationships.

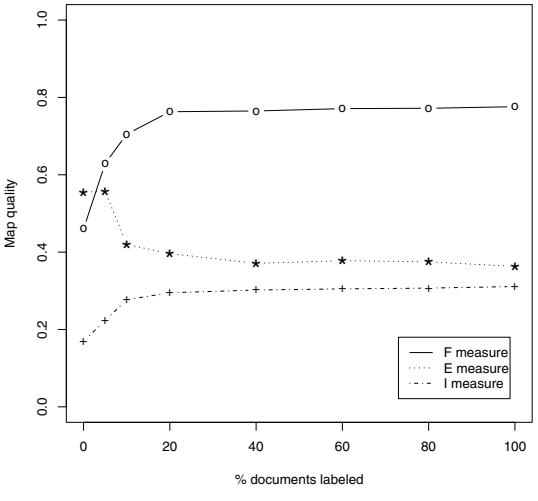


Fig. 4. Evaluation measures for the semi-supervised Torgerson MDS algorithm with average similarity when the percentage of documents labeled range from 0% to 100%

Figure 4 shows the evaluation measures when the percentage of documents labeled range from 0% to 100%. According to this figure the quality of the word maps generated is similar whenever the percentage of documents labeled is larger than 10%. Moreover, with only 5% of documents categorized the performance is not significantly degraded. Finally, we report that the semi-supervised MDS algorithms with only 10% of documents categorized improve significantly the unsupervised counterparts (0% of documents labeled).

5 Conclusions and Future Research Trends

In this paper we have proposed a semi-supervised version of the Torgerson MDS algorithm for textual data analysis. The new model takes advantage of a categorization of a subset of documents to improve the discriminant power of the word maps generated. The algorithm proposed has been tested using a real textual collection and evaluated through several objective functions.

The experimental results suggest that the proposed algorithm improves significantly well known alternatives that rely solely on unsupervised measures. In particular the overlapping among different topics in the map is significantly reduced improving the discriminant power of the algorithms.

Future research will focus on the development of new semi-supervised clustering algorithms.

Acknowledgements. Financial support from Junta de Castilla y León grant PON05B06 is gratefully appreciated.

References

1. Aggarwal, C.C., Gates, S.C., Yu, P.S.: On Using Partial Supervision for Text Categorization. IEEE Transactions on Knowledge and Data Engineering 16(2), 245–255 (2004)
2. Aggarwal, C.C.: Re-designing distance functions and distance-based applications for high dimensional applications. In: Proc. of SIGMOD-PODS, vol. 1, pp. 13–18 (2001)
3. Bartell, B.T., Cottrell, G.W., Belew, R.K.: Latent Semantic Indexing is an Optimal Special Case of Multidimensional Scaling. In: ACM SIGIR Conference, Copenhagen, Denmark, pp. 161–167 (1992)
4. Berry, M.W., Drmac, Z., Jessup, E.R.: Matrices, vector spaces and information retrieval. SIAM review 41(2), 335–362 (1999)
5. Bock, H.H.: Simultaneous visualization and clustering methods as an alternative to Kohonen maps. In: Della Riccia, G., Kruse, R., Lenz, H.-J. (eds.) Learning, networks and statistics, CISM Courses and Lectures no. 382, pp. 67–85. Springer, Wien - New York (1997)
6. Buja, A., Logan, B., Reeds, F., Shepp, R.: Inequalities and positive default functions arising from a problem in multidimensional scaling. Annals of Statistics 22, 406–438 (1994)
7. Chang, H., Yeung, D.-Y., Cheung, W.K.: Relaxational Metric Adaptation and its Application to Semi-Supervised Clustering and Content-Based Image Retrieval. Pattern Recognition 39, 1905–1917 (2006)
8. Chapelle, O., Weston, J., Schölkopf, B.: Cluster kernels for semi-supervised learning. In: Conference on Neural Information Processing Systems (NIPS), vol. 15 (2003)
9. Chen, H., Houston, A.L., Sewell, R.R., Schatz, B.R.: Internet browsing and searching: User evaluations of category map and concept space techniques. Journal of the American Society for Information Science (JASIS) 49(7), 582–603 (1998)
10. Cox, T.F., Cox, M.A.A.: Multidimensional scaling, 2nd edn. Chapman & Hall/CRC.ition, USA (2001)

11. Joachims, T.: Learning to Classify Text using Support Vector Machines. In: Methods, Theory and Algorithms, Kluwer Academic Publishers, Boston (2002)
12. Kaufman, L., Rousseeuw, P.J.: Finding groups in data. In: An introduction to cluster analysis, John Wiley & Sons, New York (1990)
13. Kohonen, T.: Self-organizing maps, 2nd edn. Springer, Berlin (1995)
14. Kothari, R., Jain, V.: Learning from Labeled and Unlabeled Data Using a Minimal Number of Queries. IEEE Transactions on Neural Networks 14(6), 1496–1505 (2003)
15. Lebart, L., Salem, A., Berry, L.: Exploring Textual Data. Kluwer Academic Publishers, Netherlands (1998)
16. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
17. Mao, J., Jain, A.K.: Artificial neural networks for feature extraction and multivariate data projection. IEEE Transactions on Neural Networks 6(2) (March 1995)
18. Martín-Merino, M., Muñoz, M.: A New MDS Algorithm for Textual Data Analysis. In: Pal, N.R., Kasabov, N., Mudi, R.K., Pal, S., Parui, S.K. (eds.) ICONIP 2004. LNCS, vol. 3316, pp. 860–867. Springer, Heidelberg (2004)
19. Martín-Merino, M., Muñoz, A.: A New Sammon Algorithm for Sparse Data Visualization. Int. Conf. on Pattern Recognition 1, 477–481 (2004)
20. Mladenié, D.: Turning Yahoo into an Automatic Web-Page Classifier. In: Proceedings of the 13th European Conference on Aritficial Intelligence, Brighton, pp. 473–474 (1998)
21. Pedrycz, W., Vukovich, G.: Fuzzy Clustering with Supervision. Pattern Recognition 37, 1339–1349 (2004)
22. Strehl, A., Ghosh, J., Mooney, R.: Impact of similarity measures on web-page clustering. In: Proceedings of the 17th National Conference on Artificial Intelligence: Workshop of Artificial Intelligence for Web Search, Austin, USA, July 2000, pp. 58–64 (2000)
23. Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, New York (1998)
24. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proc. of the 14th International Conference on Machine Learning, Nashville, Tennessee, USA, July 1997, pp. 412–420 (1997)

Landscape Multidimensional Scaling

Katharina Tschumitschew[1], Frank Klawonn[1], Frank Höppner[2], and Vitaliy Kolodyazhniy[3]

[1] Department of Computer Science
University of Applied Sciences Braunschweig/Wolfenbüttel
Salzdahlumer Str. 46/48
D-38302 Wolfenbuettel, Germany
{katharina.tschumitschew,f.klawonn}@fh-wolfenbuettel.de
[2] Department of Economics
University of Applied Sciences Braunschweig/Wolfenbüttel
Robert Koch Platz 10-14
D-38440 Wolfsburg, Germany
f.hoeppner@fh-wolfenbuettel.de
[3] Institute for Psychology
University of Basel
Missionsstrasse 60/62
Basel, CH-4055, Switzerland
v.kolodyazhniy@unibas.ch

Abstract. We revisit the problem of representing a high-dimensional data set by a distance-preserving projection onto a two-dimensional plane. This problem is solved by well-known techniques, such as multidimensional scaling. There, the data is projected onto a *flat plane* and the Euclidean metric is used for distance calculation. In real topographic maps, however, travel distance (or time) is not determined by (Euclidean) distance alone, but also influenced by map features such as mountains or lakes. We investigate how to utilize *landscape features* for a distance-preserving projection. A first approach with rectangular cylindrical mountains in the MDS landscape is presented.

Keywords: visualisation, multidimensional scaling, landscape multidimensional scaling.

1 Introduction

Large data sets call for tools that (semi-) automate as many of the tedious steps in data analysis as possible, but usually cannot replace the visual inspection of data or results, because the visual perception of humans is extremely good in detecting abnormalities that are difficult to detect automatically. This explains why visualisation techniques are useful and important, even in times of powerful data mining techniques and large data sets.

In this paper, we therefore revisit the problem of mapping high-dimensional data to a two- or three-dimensional representation. There are various approaches

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 263–273, 2007.

to solve this problem: a data record may be represented by a pictogram, where each attribute is mapped to a detail of the pictogram (e.g. stick figures or Chernoff faces), a record may be represented by a sequence of line segments (e.g. one line segment per attribute as with parallel coordinates), or it may be projected into a low-dimensional space and then represented by a dot in a scatter plot. In the latter category the main objective is usually to preserve either the variance of the data (principal component analysis (PCA) [4]) or the distances between the data objects (e.g. multidimensional scaling (MDS) [7], and modifications thereof [8]). For most of these techniques the graph consists of one graphical element per data record (pictogram, sequence of lines, dot). Only with very few visualisation techniques additional elements provide further information about the depicted data. Just like in a map with level curves, where the existence of mountains between two geographical positions indicate a longer traveltime, we want to understand the graph as a landscape that carries information in its own right.

The paper is organized as follows: In section 2 we discuss the kind of landscape we will consider and justify it by some theoretical considerations. We stick to the idea of a distance-preserving map and, starting from a flat map (discussed in section 3), we incrementally modify the landscape to improve the overall error (section 4). Some results and examples are provided in section 5.

2 MDS Representation on a Landscape

We assume that a p-dimensional data set $X = \{x_1, x_2, ..., x_n\} \subseteq \mathbb{R}^p$ is given. By $d(v, w)$ we denote the Euclidean distance $\|v - w\|$. In MDS each of the high-dimensional data points x_i has to be mapped to a low-dimensional representative y_i. The projection of X is denoted as $Y = \{y_1, y_2, ..., y_n\} \subseteq \mathbb{R}^q$ where $1 \leq q < p$ (typically $q \in \{2, 3\}$). A perfect distance-preserving projection of X to Y would keep the distances $d^x_{ij} := d(x_i, x_j)$ of the high-dimensional space identical to the distances $d^y_{ij} := d(y_i, y_j)$ of the projected data objects, that is, $d^x_{ij} = d^y_{ij}$ holds. A perfect projection is, however, impossible except for a few trivial cases. Therefore, MDS seeks to minimize the error introduced by the projection ($|d^x_{ij} - d^y_{ij}|$ for all i, j). Common objective functions are [7]:

$$E_1 = \frac{1}{\sum_{i=1}^{n} \sum_{j=i+1}^{n} \left(d^x_{ij}\right)^2} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left(d^y_{ij} - d^x_{ij}\right)^2, \tag{1}$$

$$E_2 = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left(\frac{d^y_{ij} - d^x_{ij}}{d^x_{ij}} \right)^2, \tag{2}$$

$$E_3 = \frac{1}{\sum_{i=1}^{n} \sum_{j=i+1}^{n} d^x_{ij}} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \frac{\left(d^y_{ij} - d^x_{ij}\right)^2}{d^x_{ij}}. \tag{3}$$

Usually the selected *stress function* is minimized by a numerical optimisation method such as gradient descent.

A similar technique that does not use any of the above objective functions is the self-organizing map (SOM) [6]. The data objects are assigned to cells on a regular grid such that data objects in neighbouring cells are similar to each other. Traditionally, the colouring of the cells is used to provide additional information about the similarity to data in adjacent cells. In [9] an extension has been proposed that encodes information about the data density as well as the distance between cell data in a *landscape*, the so-called U^*-matrix. The distance of neighbouring cells is reflected by the landscape, but for non-adjacent cells no conclusions can be made.

In this work we stick to a distance-preserving map (just as with MDS), but we want to use the landscape as an additional parameter influencing the *perceived distance* between data objects placed in the landscape. With a real map the true travel distance depends on the chosen path: we may circumvent or climb a mountain, for instance. To be of immediate use no tedious *path optimisation* should be necessary to understand the visualisation, therefore only the straight connection between the points is considered relevant for their distance. If the straight line between two projected points is shorter than the distance between their high-dimensional originals, we may introduce obstacles on the path to increase their map-distance. From the number and height of the obstacles a user gets a better impression of the true distance.

A landscape MDS (LMDS) may be constructed in the following way: We seek for an initial distribution of data objects on a flat plane guaranteeing that

$$d_{ij}^y \leq d_{ij}^x \tag{4}$$

holds for all $i, j \in \{1, \ldots, n\}$. This condition is motivated by the fact that mountains can only increase the distances. Then we place rectangular or cylindrical mountains (parameterized by location, size and height) in the landscape such that the difference $|d_{ij}^y - d_{ij}^x|$ is reduced. In our simple first model a mountain increases the path length by twice the height of the cylindrical mountain, corresponding to climbing the mountain and descending to the base level again.

Can such a *landscape MDS* deliver better results than traditional MDS? Yes, it can, at least in theory. We place the projections y_i on a circle such that the distances are no larger than the original ones and no three points are collinear (figure 1, left). We ensure $d_{ij}^y \leq d_{ij}^x$ by making the circle sufficiently small. Then introduce cylindrical mountains m_{ij} $(i < j)$ such that each mountain m_{ij} is crossed only by the line connecting the two points y_i and y_j (figure 1, right). Choosing $h_{ij} = (d_{ij}^x - d_{ij}^y)/2$ reduces the error of the distances to zero in the landscape.

The drawback of this solution is obviously that in the worst case $n(n-1)/2$ very narrow cylindrical mountains for n data objects have to be introduced. The resulting landscape is very different from common maps and thus hard to interpret. We will develop a better alternative in the following sections.

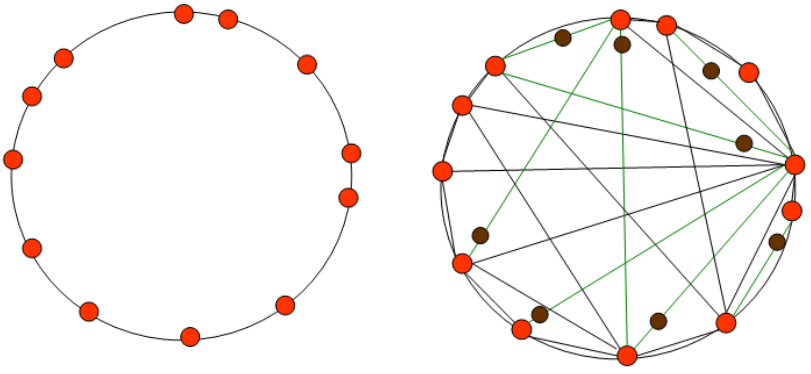


Fig. 1. An optimal solution with zero error

3 Initialisation of LMDS

As mentioned above, landscape MDS (LMDS) must be initialized such that the constraints (4) for the distances are fulfilled. There are a number of approaches which can be used for the initialization of an LMDS map. To choose the best initialization strategy, we performed a comparison of the following four approaches:

(a) Projecting the data to the plane by PCA approach automatically satisfies the constraint (4) of smaller distances.
(b) The second approach enhances the first one by shifting points within allowed intervals after the PCA projecting. The intervals are computed such that the error function (3) is minimized and the violation of the constraints (4) is avoided. The interval computation involves the golden section search algorithm.
(c) Classical MDS which violates the constraint of smaller distances.
(d) MDS with constraints on distances using a Lagrange function which is defined as

$$L = E_3 + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \lambda_{ij} \left(d_{ij}^y - d_{ij}^x \right). \tag{5}$$

Although no analytical solution of (5) exists, the Lagrange function can be optimized iteratively using the Arrow-Hurwitz-Uzawa algorithm [2]:

$$\begin{cases} \lambda_{ij}^{(k+1)} & = \max\left\{0,\ \lambda^{(k)} + \alpha^{(k)} \left(d_{ij}^y - d_{ij}^x \right) \right\}, \quad i,j = 1,\ldots,n \\ y_i^{(k+1)} & = y_i^{(k)} - \beta^{(k)} \frac{\partial L\left(y^{(k)}, \lambda^{(k)}\right)}{\partial y_i}, \qquad\qquad i = 1,\ldots,n \\ 0 < \alpha^{(k)} \leq 1 & \\ 0 < \beta^{(k)} \leq 1 & \end{cases} \tag{6}$$

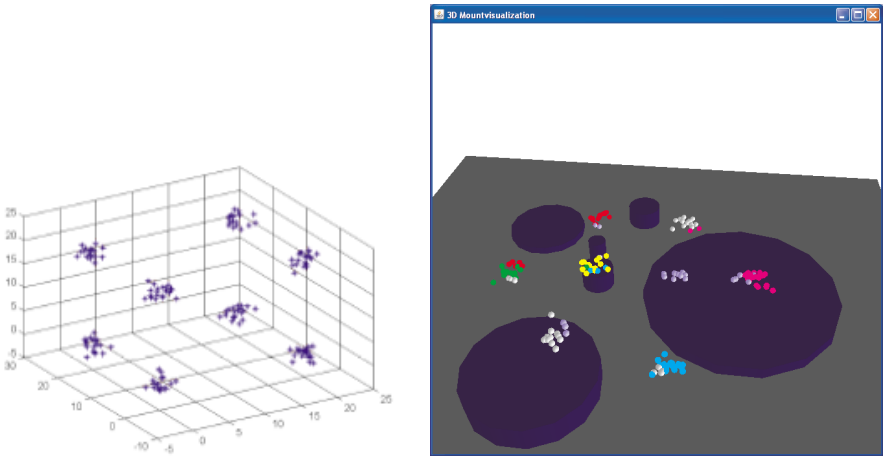


Fig. 2. The cube data set

where $\alpha^{(k)}$ and $\beta^{(k)}$ are the step length parameters. The undetermined Lagrange multipliers $\lambda_{ij}^{(k+1)}$ are equal to zero when the constraints (4) are satisfied, and become positive when the constraints are violated.

To compare the initialization results provided by the different approaches listed above, we used a number of benchmark data sets: 'Iris' (4 attributes) and 'Glass' (9 attributes) from the UCI Machine Learning Repository [1] with 150 and 214 data objects, respectively, as well as the two three-dimensional data sets 'Cube' and 'Coil' (see the left images in figures 2 and 3, respectively.) with 360 and 160 data objects, respectively. The results for these data sets provided by the considered initialization methods are listed in table 1. The number of iterations required for PCA with shifting was 5, because more iterations did not significantly improve the results. For constrained MDS with the Arrow-Hurwitz-Uzawa algorithm, the number of iterations was 5000.

As can be seen from table 1, constrained MDS with the Arrow-Hurwitz-Uzawa optimisation procedure considerably outperforms the other three approaches w.r.t. the objective function (3) and avoiding the violation of constraints (4). So we will use this technique for the initialization of LMDS in the following section. The important advantage of the procedure (6) are its high speed of convergence and its low computational costs.

4 Adding Mountains to the Landscape

In the previous section, initialisation strategies were compared that position the data points on a flat landscape in such a way that the constraints (4) are (almost) satisfied. Therefore, in the next step, mountains will be introduced that can only lead to in an increase of the distances of the flat landscape. The effect

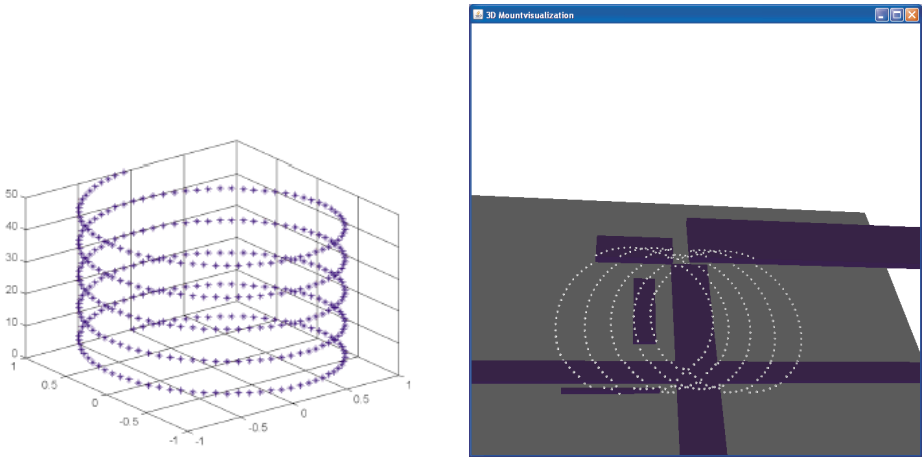


Fig. 3. The coil data set

Table 1. Empirical comparison of the initialisation strategies

Data set	PCA: Error (3)	MDS: Error (3) Sum of violations of constraint (4)	PCA with shifting: Error (3) Sum of violations of constraint (4)	Arrow-Hurwitz-Uzawa MDS: Error (3) Sum of violations of constraint (4)
Iris	0.0097589 0	0.00632271946452 453.0641	0.009536704494 0	0.0090821097634 3.085490787E-4
Coil	0.0768078 0	0.0546562810427 8805.257	0.076738333 0	0.0746009358153 8.20138002E-4
Glass	0.170403 0	0.03577761897878 4176.65	0.10563177615 0	0.080108479955 7.677778033E-4
Cube	0.070009 0	0.0479180387553 1669.474	0.067812491266 0	0.0652975311919 7.673804646E-4

of a cylindrical or rectangular mountain of height h on the distance between two points i and j is

$$d_{ij}^{new} = d_{ij} + \xi_{ij} h, \qquad \xi_{ij} \in \{0,\ 1,\ 2\}\,. \tag{7}$$

Figure 4 shows the possible cases and the values for ξ_{ij}.

When we add mountains to the landscape, we have to decide where to place the mountains (the underlying rectangle for a rectangular mountain and centre point as well as the radius for a cylindrical mountain), determine their heights and also the number of mountains we want to introduce. Again, there is no analytical solution to this problem. The objective function is not even continuous according the coefficients ξ_{ij}. Therefore, we apply an evolution strategy (see

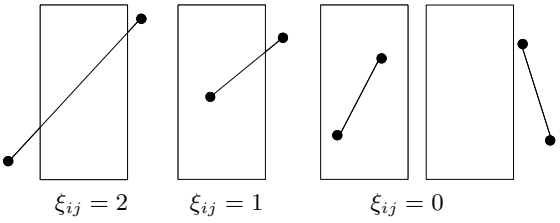


Fig. 4. Effects of a rectangular mountain on the distance between two points

for instance [3]) for positioning the mountains. The height of a mountain is not considered as a parameter of the evolution strategy, since the optimal height for a mountain with a fixed location can be obtained from $\frac{\partial E_3}{\partial h} = 0$, leading to

$$h \;=\; -\frac{\sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{d_{ij}^{y}-d_{ij}^{x}}{d_{ij}^{x}}\xi_{ij}}{\sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{\xi_{ij}}{d_{ij}^{x}}\xi_{ij}}. \tag{8}$$

There are two strategies to add mountains to the landscape. Mountains can be introduced to the landscape one by one or a fixed number of mountains is added simultaneously. In both cases, the intersection of mountains must be avoided. Otherwise, (7) would not be valid anymore and the computing of the revised distances would become quite complicated. It is, however, allowed that one mountain is placed completely on top of another, as long as there is not just a partial overlap of mountains. When k moutains are added simultaneously and not one by one to the landscape, the optimal heights of the mountains can no longer be computed from (8). The heights are given by the solution of the following system of linear equations.

$$\sum_{l=1}^{k} h_l \sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{\xi_{ijl}}{d_{ij}^{x}}\xi_{ijp} \;=\; -\sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{d_{ij}^{y}-d_{ij}^{x}}{d_{ij}^{x}}\xi_{ijp} \qquad (p = 1,\ldots,k) \tag{9}$$

The solution of this system of linear equations can lead to negative heights. Negative heights as well as the avoidance of overlapping mountains is enforced by a dynamic penalty function to the evolution strategy that assigns a low fitness to solutions with intersecting mountains. The penalty function for cylindrical mountains is described here. The corresponding penalty function for rectangular mountains can be defined in a similar way. Two cylindrical mountains with radius r_i and r_j overlap when the distance $\delta^{(ij)}$ between the midpoints of their circles satisfies

$$\delta^{(ij)} \;\leq\; r^i + r^j. \tag{10}$$

If in this case either $\delta^{(ij)} + r^i \leq r^j$ or $\delta^{(ij)} + r^j \leq r^i$ holds, then one of the mountains lies completely on top of the other and no penalty is required. In case

a penalty is needed, it is defined as $g_{ij} = r^i + r^j - \delta^{(ij)}$, otherwise $g_{ij} = 0$ is chosen. The overall penalty is given by

$$\text{pen} = \sum_{i=1}^{k} -\min\{h_i, 0\} + \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} g_{ij}. \tag{11}$$

The penalty function is increased with the number of generations of the evolution strategy, so that in the end solutions with intersecting mountains have no chance to survive due to the bad fitness value. The overall fitness is given equation (3) plus

$$f(t) \cdot \text{pen} \tag{12}$$

where $f(t)$ is a positive and increasing function. t refers to the generation number of the actual population of the evolution strategy.

In order to speed up the computations, the determination of the ξ_{ij} values for rectangular mountains is carried out based on the Cohen-Sutherland line-clipping algorithm (see for instance [5]) used in computer graphics. Line clipping refers to the problem of finding out whether a line has to be drawn in a rectangular window on the computer screen, so that it is necessary to determine, whether the line is completely inside, completely outside or partly inside the window. This corresponds exactly to the problem of finding out whether a connecting line between two points is affected by a rectangular mountain, as it is illustrated in figure 4.

For cylindrical mountains the same strategy is used by applying the Cohen-Sutherland line-clipping algorithm to the bounding square of the circle associated with the cylindrical mountain. In case a line intersects the bounding square, it is still necessary to test whether it intersects also the circle. But at least for lines outside the bounding square, we know $\xi_{ij} = 0$.

Introducing mountains in a greedy manner step by step with the evolution strategy turned out to be faster than adding mountains simultaneously. However, the results of the latter strategy were slightly better in our experiments.

5 Examples

In this section we present the results of numerical simulations of the proposed LMDS visualisation method. We used five data sets: the first four are the artificial data sets 'Cube', 'Pyramid', 'Coil' and 'Ring'. Two of them were already mentioned in section 3. The third data set is a real-world data set from a wastewater treatment (WWT) plant in Germany. The results are shown in figures 2, 3, 5, 6 and 7, respectively. Note that the 3D-effect is better, when the images can be rotated or zoomed. The four artificial data sets are all three-dimensional. On the left-hand side of each figure, the original data set is shown, on the right-hand side the LMDS result.

For the WWT plant, measurements of 15 process variables over a period of six years were available. The main task was to visualize the year-to-year variations

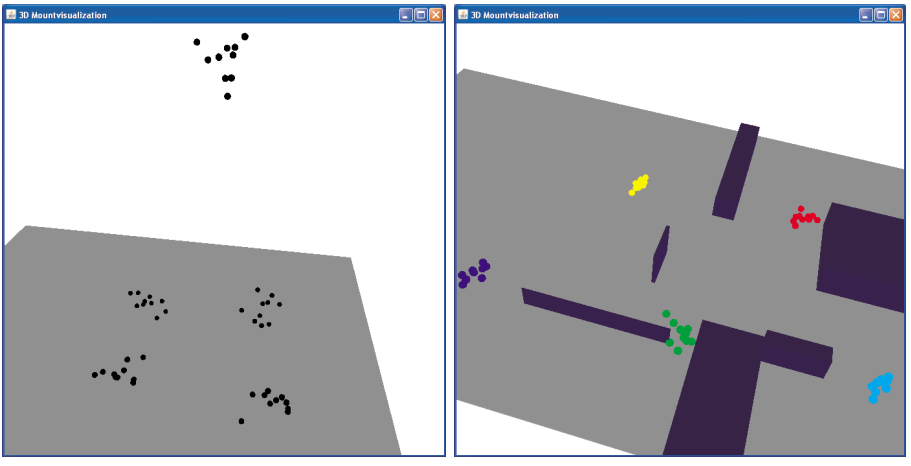


Fig. 5. The pyramid data set

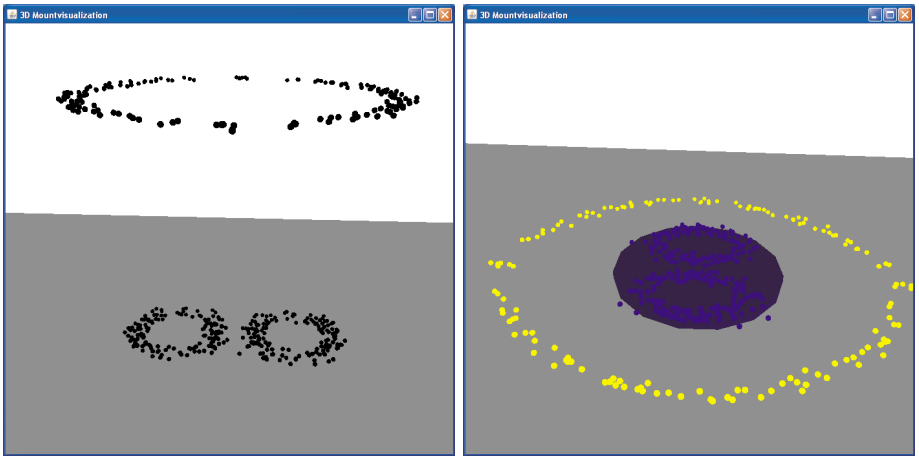


Fig. 6. The ring data set

in the plant. We did not use any information on time or date for the generation of the visualisation.

The 3D-diagram generated from our method shows a clear separation of the year 1996 which is marked by bigger yellow spheres. This is confirmed by our knowledge of changes in the process that were implemented in the year of 1997.

Table 2 shows a comparison of the error values for classical MDS and for LMDS for the objective function (3). The error value of LMDS is not always better. The main reason is the greedy strategy to start with an MDS initialisation with the constraints described in equation (4). It seems that the restricted

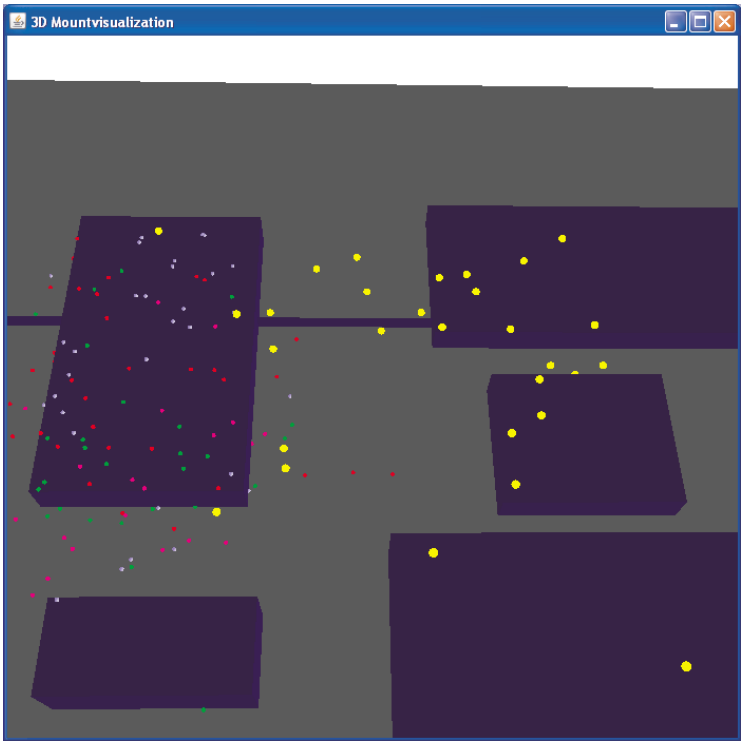


Fig. 7. The wastewater treatment plant data set

Table 2. Comparison of MDS to LMDS

Error according to equation (3)		
Data set	**MDS**	**LMDS**
Pyramid	0.02028127	0.00872146
Ring	0.02551469	0.02430803
Cube	0.04791804	0.03802418
Coil	0.05465628	0.05809677
Wastewater treatment plant	0.07254737	0.08211769

MDS gets stuck in a local minimum easier and the introduction of mountains afterwards cannot always compensate this effect completely.

6 Conclusions

We have introduced a new method that exploits a landscape for multidimensional scaling instead of a flat plane. In principle, it is possible to position the data points on the landscape in such a way that the distances in the original

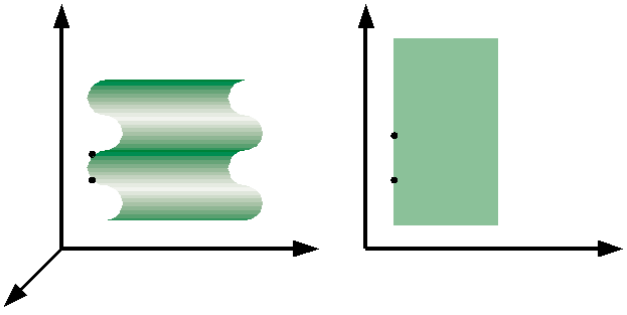


Fig. 8. The problem of fitting a lower into a higher dimensional space

high-dimensional data space are preserved exactly. However, this will lead to extremely complicated and non-intuitive landscapes.

Simplified landscapes allowing only a strictly limited number of mountains suitable for real visualisation purposes cannot guarantee this exact representation of the distances anymore. However, shortening distances instead of increasing them by mountains might lead to better solutions. The reason for this problem can be seen in figure 8. The typical way to fit a two-dimensional plane into a higher dimensional space would be to fold it like a towel, which would lead to larger distances instead of smaller distances as illustrated in figure 8. But mountains in LMDS can only increase, but not decrease distances and concepts like 'wormholes' that would be needed to shorten distances are not very suitable for visualisation purposes.

References

1. UCI machine learning repository, www.ics.uci.edu/~mlearn/MLRepository.html
2. Arrow, K., Hurwitz, L., Uzawa, H.: Studies in Nonlinear Programming. Stanford University Press, Stanford (1958)
3. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, Oxford (1996)
4. Dunn, G., Everitt, B.: An Introduction to Mathematical Taxonomy. Cambridge University Press, Cambridge, MA (1982)
5. Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F.: Computer Graphics: Principles and Practice. 2nd edn. in C. Addison-Wesley, Boston (1996)
6. Kohonen, T.: Self organizing maps. Springer, Heidelberg, New York (2001)
7. Kruskal, J.B., Wish, M.: Multidimensional Scaling. SAGE Publications, Beverly Hills (1978)
8. Rehm, F., Klawonn, F., Kruse, R.: MDS_{polar}: A new approach for dimension reduction to visualize high dimensional data. In: Famili, A.F., Kook, J.N., Peña, J.M. (eds.) Advances in Intelligent Data Analysis V, pp. 316–327. Springer, Berlin (2005)
9. Ultsch, A.: Clustering with SOM: U*C. In: Workshop on Self-Organizing Maps (WSOM 2005), Paris, pp. 75–82 (2005)

A Support Vector Machine Approach to Dutch Part-of-Speech Tagging

Mannes Poel, Luite Stegeman, and Rieks op den Akker

Human Media Interaction, Dept. Computer Science, University of Twente,
P.O. Box 217, 7500 AE Enschede, The Netherlands
{mpoel,infrieks}@ewi.utwente.nl
http://hmi.ewi.utwente.nl/

Abstract. Part-of-Speech tagging, the assignment of Parts-of-Speech to the words in a given context of use, is a basic technique in many systems that handle natural languages. This paper describes a method for supervised training of a Part-of-Speech tagger using a committee of Support Vector Machines on a large corpus of annotated transcriptions of spoken Dutch. Special attention is paid to the decomposition of the large data set into parts for common, uncommon and unknown words. This does not only solve the space problems caused by the amount of data, it also improves the tagging time. The performance of the resulting tagger in terms of accuracy is 97.54 %, which is quite good, where the speed of the tagger is reasonably good.

Keywords: Part-of-Speech tagging, Support Vector Machines

1 Introduction

In a text, every word belongs to a certain word class, such as noun or verb. There are many more classes and classes can often be subdivided into smaller classes, for example by taking tense, number or gender into account. Some words belong only to a single word class. For example the word *jaar* (year), which is strictly a singular noun. Many words, however, belong to more than one word class, the so-called *ambiguous* words. The actual class of such a word depends on the context. The more common a word is, the more likely it is to be ambiguous. In the Corpus Gesproken Nederlands (CGN - Spoken Dutch Corpus) a large morpho-syntactically annotated corpus that we used for the experiments in this paper, the percentage of ambiguous words related to the number of occurrences in the corpus is shown in Figure 1 (details of this corpus are given in Section 1.1). The data points are an average of the percentage of ambiguous words around that frequency. Part-of-Speech (PoS) tagging, or word class disambiguation, is the process of finding out the right word class for these ambiguous words. The result is then added as a label or tag to the word. PoS tagging is often only one step in a text processing application. The tagged text could be used for deeper analysis, for example for chunk parsing or full parsing. Because the accuracy of the PoS tagging greatly influences the performance of the steps further in the pipeline [1], the accuracy of the PoS tagger is very important.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 274–283, 2007.

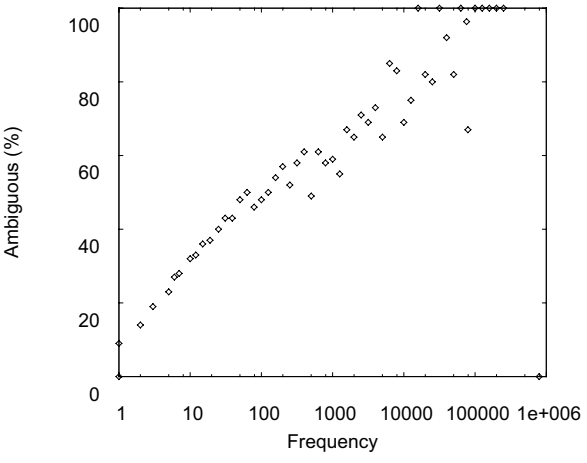


Fig. 1. The average percentage of ambiguous words related to the number of occurrences in the CGN corpus

Various models for supervised machine learning have been applied to the problem of PoS tagging; memory based learning [2] , transformation rule based learning [3], (Hidden) Markov Models. Apart from the overall accuracy, relevant measures for PoS taggers concern the accuracy of handling unknown words, the amount of training data required (the learning rate), training time, tagging time, and the accuracy on different types of corpora. TnT, a trigram HMM tagger by Brants [4], has shown good results in both accuracy and training time as well as tagging time.

During the development of the CGN corpus, a number of methods for PoS tagging have been compared and used for bootstrapping. Zavrel and Daelemans [5] report on a number of PoS taggers trained and tested on the CGN. Canisius and van den Bosch [1] used a subset of the CGN for learning system to find basic chunks (i.e. sentential phrases, such as noun phrases and prepositional phrases).

The aim of this research is to find the appropriate ingredients for constructing a PoS tagger for spoken Dutch, using Support Vector Machines (SVMs) [6], a classification method well known for its good generalization performance [7].

1.1 Spoken Dutch Corpus (CGN)

The Spoken Dutch Corpus (Corpus Gesproken Nederlands, CGN) [8] is a database of contemporary spoken Dutch. It consists of almost 9 million transcribed words of spoken Dutch, divided into 15 different categories, cf. Table 1. Of these words, around two thirds originate from the Netherlands, the remaining one third from Flanders. The entire CGN is annotated with a large set of PoS tags. The full set consists of 316 different tags, which denote many different features of a word class. An example of such a tag is N(soort,ev,basis,zijd,stan), meaning noun, sort name,

Table 1. The 15 different categories of the Dutch Spoken Corpus (CGN)

Category	Type	Size in words
A	Face to face conversations	2626172
B	Interview with teacher Dutch	565433
C	Phone dialogue (recorded at platform)	1208633
D	Phone dialogue (recorded with mini disc)	853371
E	Business conversations	136461
F	Interviews and discussions recorded from radio	790269
G	Political debates, discussions and meetings	360328
H	Lectures	405409
I	Sport comments	208399
J	Discussions on current events	186072
K	News	368153
L	Comments on radio and TV	145553
M	Masses and ceremonies	18075
N	Lectures and discourses	140901
O	Text read aloud	903043

singular, basis (not diminutive), not neuter, standard case (not dative or genitive). A full explanation of the features and their use can be found in [9]. Many of the pronouns contain even more features, up to nine. This subdivision is so fine-grained that many tags occur only a few times in the entire corpus. There are even 19 tags that occur only once. Although it is possible to discard all the subclasses and use only the main class, this would leave us with a set of only 12 tags (including LET and SPEC, for punctuation mark and special respectively). A tag set of this size is much smaller than what is commonly used in PoS tagging. Discarding all these features also reduces the value of the tagged data to further processing steps. To overcome this problem, the syntactic annotations use a different tag set, consisting of 72 tags. These tags are a reduced version of the full tag set, making it more suitable for machine learning. Only about ten percent of the corpus is tagged using these tags, but the tags can be automatically derived for the rest of the corpus using a set of simplification rules and the full tags. Table 2 shows an overview of this tag set. The challenges of using SVMs on such a large corpus as the CGN is the size of the training set which requires a separation of the corpus, and to find a workable representation of the input. We will show in this paper how we solved these problems by decomposition and by using a committee of modular SVMs.

2 Design of the SVM Tagger

Several successful SVM PoS taggers can already be found in the literature, see for instance the work of Giménez and Márquez [10]. They constructed accurate SVM PoS taggers for English and Spanish . In their approach a linear kernel was used. Nakagawa, Kudo and Matusmoto [11] constructed a polynomial kernel SVM PoS tagger for English. However both of the above approaches are applied

Table 2. Tags in the medium-sized tag set of size 72. The items in the second column are the main classes and correspond to the reduced tag set of 12 tags.

Tag numbers	Part-of-Speech Tag	Tags in the CGN corpus
1 . . . 8	Noun	N1, N2, . . ., N8
9 . . . 21	Verb	WW1, WW2, . . ., WW13
22	Article	LID
23 . . . 49	Pronoun	VNW1, VNW2, . . ., VNW27
50, 51	Conjunction	VG1, VG2
52	Adverb	BW
53	Interjections	TSW
54 . . . 65	Adjective	ADJ1, ADJ2, . . ., ADJ12
66 . . . 68	Preposition	VZ1, VZ2, VZ3
69, 70	Numeral	TW1, TW2
71	Punctuation	LET
72	Special	SPEC

to written text only and are applied to a corpus of a much smaller size than the CGN corpus.

In this research we opted for a single pass left-to-right tagger using a sliding window consisting of 7 consecutive words in which the tagger needs to determine the tag of the middle word; word 4.

The first stage in the design is the transformation of words to numerical values.

2.1 Input Coding

As stated above the input of the tagger consists of a sliding window of 7 words. Each of the 7 words in this window is encoded based on the coding description given in Table 3. Many of the features in Table 3 are so-called "1-out-of-N" encodings. For example the *word* encoding is a vector of length equal to the size of the lexicon with only a 1 at the index of the word. Special attention has to be paid to unknown words, words that have no tag frequency data available. For context words, the average relative frequencies for unknown words are used, cf. Figure 2. It follows from Figure 2 that only a few tags have unknown words. For the unknown word to be tagged, all frequency values are set to zero.

It should be stated that in the testing phase the PoS tag for the preceding words is the tag generated by the SVM PoS tagger (the tagger is a left-to-right tagger).

2.2 Training and Test Data

From the CGN data set a total of 11 sets was constructed. The first sentence of the data set was put in *set0*, the second sentence in *set1*, etc. Of these sets *set0* was used for testing and *set1* up to *set10* for training. Using the input

Table 3. Input coding vector for the SVMs. The input is a sliding window of 7 words, $w1w2w3w4w5w6w7$, and $w4$ is the word to be tagged (PoST means Part-of-Speech Tag).

Type	Used for	Coding
PoST	w1, w2, w3	"1-out-of-N"
relative tag frequencies	w4, w5, w6, w7	vector of relative tag frequencies
suffix	w4	"1-out-of-N"
word	w1, ..., w7	"1-out-of-N"
capitalization	w1, ..., w7	0 for no capitals, 1 for the first letter, 2 for more then one letter
length	w1, ..., w7	single number
number	w1, ..., w7	0 if word does not contain number, 1 if it contains at least one number, 2 if the first character is a number, 3 if all characters are numbers
suffix	w4	"1-out-of-N"
PoST bigrams	w1, w2, w3	"1-out-of-N"
PoST trigrams	w1, w2, w3	"1-out-of-N"
Reduced PoST bigrams	w1, w2, w3	"1-out-of-N"
Reduced PoST trigrams	w1, w2, w3	"1-out-of-N"
word bigrams	w1, w2, w3, w5, w6, w7	"1-out-of-N"

coding described in Section 2.1, this resulted in a training set of more than 2 Gigabytes, which is more than the system could handle. One option is to reduce the training set by taking a random sample, but this increases the probability of an unknown word during testing and hence the performance of the tagger will decrease. Therefore we opted for a modular approach which allows for a decomposition of the training data.

2.3 Decomposing the SVM Part-of-Speech Tagger

A committee of SVM taggers was designed in order to decompose the overall tagger such that for each tagger there is a reasonable amount of training data. Recall that the input of the tagger is a window of size 7; it consists of 7 consecutive words in the sentence, say $w1w2w3w4w5w6w7$ and $w4$ is the word to be tagged.

If $w4$ is a common word, a word occurring often (more than 50 times) in the training data, then there is enough training data for each separate word. Hence for each common word a single multi-class SVM is trained. For each such SVM the relative tag frequency of $w4$ is constant and can be discarded from the input coding.

If $w4$ is an uncommon word then a SVM is selected based on the reduced tag of $w3$. Since there are only 12 reduced tags, this results in enough but not

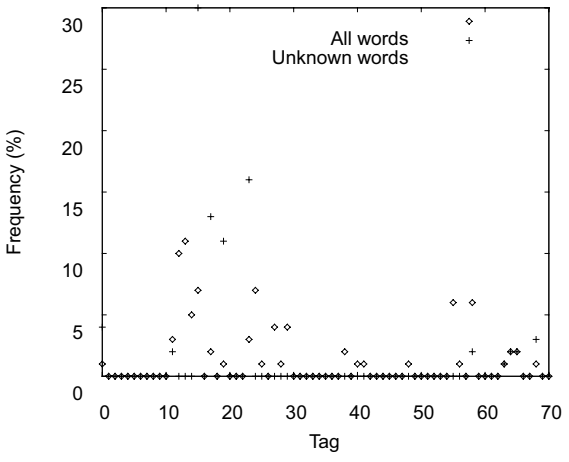


Fig. 2. The average tag frequencies for (unknown) words

too much training data. So 12 separate multi-class SVMs are trained for the uncommon words.

If $w4$ is an unknown word then again a SVM is selected based on the reduced tag of $w3$, resulting in 12 multi-class SVMs for the unknown words. Recall that if $w4$ is an unknown word then the relative tag frequency is set to zero and thus constant. Hence the relative tag frequency of $w4$ can be discarded from the input coding for these SVMs.

The selection of which tagger of the committee should classify the data is based on the properties of the word to be tagged. Given the input $w1w2w3w4w5w6w7$ the selection procedure is as follows:

if $w4$ = common word **then**
  select *SVM-common*($w4$)
else-if $w4$ = uncommom word **then**
  select *SVM-uncommon*(reduced_tag($w3$))
else $w4$ is unknown word
  select *SVM-unknown*(reduced_tag($w3$))

The different SVMs were implemented and trained using the LIBSVM toolkit, [12].

2.4 Kernel Optimization

Based on the findings in Table 4 it was decided to use a 3rd order polynomial kernel for all the SVMs. The 3rd order polynomial has the highest performance on the uncommon word class and among the best performance on the other classes.

Table 4. Tagging accuracy (in %) of the different kernels on the ambiguous words in the different classes; common, uncommon and unknown. In the overall performance the classification of non-ambiguous words is also taken into account. Hence the overall performance is higher than the average performance on the ambiguous components. This accuracy is based on a training set of size 10000 for common words and of size 50000 for uncommon and unknown words.

Kernel type	common	uncommon	unknown	overall (all words)
rbf	97.65	87.42	54.14	97.81
polynomial; 2nd order	97.67	87.14	53.47	97.82
polynomial; 3rd order	97.66	87.64	53.47	97.82
linear	97.65	87.25	52.90	97.81

3 Test Results

It can be seen from Table 4 that the performance on unknown words is very low, around 53%. Although there are not many unknown words due to the size of the CGN corpus, this performance on unknown words reduces the overall tagging performance. Hence a reasonable increase in performance can be gained if one succeeds in handling unknown words in a more intelligent way. This would also lead to better results when using the tagger on Dutch texts outside the CGN corpus.

The main difference between unknown words and known words is that for unknown words no tag frequency is available.

3.1 Improving Unknown Word Performance

Since many words in Dutch are compounds it is an option analyze the compound structure of unknown words. The final compound is the head part of the compound - the head noun of *fietspomp* (bike pump) is *pomp* - we will assume that the tags of the last compound gives us the best indication for the PoS tags of the whole word. Since there are word-ending morphemes such as *lijk* in *uitzonderlijk* that are equal to a Dutch noun, more sophisticated morphological analysis could be necessary. Different scenarios were tested for several compound analyzers.

The final result of the tests was that a combined approaches was the best; if a word can be compounded using the strict compound analyzer (first part of the compound must be in the lexicon) then the strict compound analyzer is applied, otherwise the relaxed compound analyzer (first part of the compound does not need to be in the lexicon) is applied. Both use the average tag frequencies over other compounds with the same second part as tag frequency for the compound. The performance of using this combination of compound analyzers can be found in Table 5. The coverage of this method is identical to the coverage of the relaxed compound analyzer and the tagging performance is in between the performance of the strict and relaxed compound analyzer. But the overall performance is higher, resulting in a performance increase of almost 14%.

Table 5. The performance of using the combined compound analyzer

coverage (%)	accuracy on compounds	accuracy on unknown words
62.23	74.46	69.27

4 Detailed Performance Evaluation

After investigating several options for kernels and handling unknown words the final SVM based committee of PoS taggers was constructed. For all SVMs in the committee a 3rd degree polynomial kernel was used and the input coded as described in Section 2.1. Moreover the combined compound analyzer was used. Using the compound analyzer resulted in the following selection procedure for the appropriate SVM.

if $w4$ = common word **then**
  select *SVM-common*($w4$)
else-if $w4$ = uncommom word **then**
  select *SVM-uncommon*(reduced_tag($w3$))
else-if compound analysis $w4$ succeeds % $w4$ is unknown word
  select *SVM-uncommon*(reduced_tag($w3$))
else % $w4$ is unknown word
  select *SVM-unknown*(reduced_tag($w3$))

This committee was tested on the test *set0* resulting in an overall performance of 97.52%, cf. Table 6. The confidence interval can be estimated using the theory described in Chapter 14 of Alpaydin [13], resulting in a 95% confidence interval of ±0.03% and 99% confidence interval of ±0.04% for the overall performance. Canisius and van den Bosch, [1] applied a memory based PoS tagger to the CGN corpus. They achieved an overall performance of 95.96% which is significantly lower than our performance.

Table 6. Tagging performance (in %) on the test set of the final committee of taggers. The overall performance also includes the non-ambiguous words.

common	uncommon	unknown	overall (all words)
97.28	**88.40**	**70.00**	**97.52**

The final tagger was also applied to the different categories in the CGN corpus, cf. Table 1. The performance on these categories can be found in Table 7. The test was performed on a balanced (with respect to the different categories) subset of *set0*. Therefore the results differ a little bit from the performance results in Table 6, which is based on the whole *set0*. The largest variation is in the unknown word performance with a minimum of 62.07% on category M (Masses and ceremonies)

Table 7. Final tagging performance (in %) on the different categories (cf. Table 1) in the CGN. The performance are based on a balanced subset of *set0*.

categories	common	uncommon	unknown	overall (all words)
A	97.25	85.52	69.80	97.55
B	96.76	85.01	74.04	97.21
C	97.94	88.06	65.97	98.15
D	97.90	88.72	65.49	98.13
E	97.37	83.04	68.52	97.67
F	96.78	87.87	72.09	97.05
G	96.93	88.05	72.41	96.63
H	96.93	89.53	70.51	97.43
I	97.31	92.99	68.72	97.48
J	97.13	91.10	82.43	97.45
K	96.76	90.45	74.44	96.89
L	96.02	86.88	73.06	96.19
M	96.28	75.00	62.07	96.03
N	95.85	88.22	74.29	96.19
O	96.66	87.91	71.64	96.69
mean	96.92	87.22	71.03	97.12
variance	0.36	17.45	22.77	0.45

and a highest unknown word performance of 82.43% on category J (Discussions on current events).

The tagging speed was around 1000 words/sec. which is much higher than the 20 words/sec of the tagger developed by Nakagawa et all. [11].

Also a preliminary evaluation was performed with the well-known TnT tagger [4]. The TnT tagger was trained and tested on a subset of this corpus, using the same tag set. The performance of the TnT tagger was around 96.0%. A more thoroughly evaluation was performed with a Neural Network based PoST. The performance if this tagger was 97.35% on *set0*, which is also significantly lower than the 97.54% of the SVM based tagger.

5 Conclusions

The main challenge of this research was to apply SVM based PoS taggers to large corpora, especially the CGN corpus. The CGN corpus consists of around 9 million words and is the largest corpus of spoken Dutch. In order to reduce the training load a committee of SVM was designed and trained. The appropriate SVM in the committee is selected on bases of simple features of the word to be tagged and the preceding word.

The developed committee of SVMs was tested on a test set consisting of around 1 million words. The overall tagging performance was 97.52% with a 99% confidence interval of $\pm 0.04\%$, cf. Table 6. This result is significantly higher than the memory-based PoS tagger developed by Canisius and van der Bosch

[1], which has a tagging performance of 95.96%. Also the designed SVM tagger outperforms the TnT tagger, which has a performance of around 96%, and a Neural Network based tagger with 97.35% performance. Moreover the developed tagger has a reasonable speed of 1000 words/sec.

One of the interesting ingredients in the developed tagger was the use of compound analysis to boost the unknown word performance. Using compound information resulted in an increase of almost 14% on unknown words. The lowest unknown word performance is 62.07% on category M (Masses and ceremonies) and a highest unknown word performance of 82.43% on category J (Discussions on current events).

References

1. Canisius, S., van den Bosch, A.: A memory-based shallow parser for spoken dutch. In: ILK/Computational Linguistics and AI, Tilburg University (2004)
2. Daelemans, W., Zavrel, J., Berck, P., Gillis, S.: Mbt: A memory-based part of speech tagger-generator. In: Proceedings of the 4th Workshop on Very Large Corpora, ACL SIGDAT (2000)
3. Brill, E.: Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. Computational Linguistics 21(4), 543–565 (1995)
4. Brants, T.: TnT – A Statistical Part-of-Speech Tagger. In: Proceedings of the 6th Applied NLP Conference (ANLP-2000), pp. 224–331 (2000)
5. Zavrel, J., Daelemans, W.: Bootstrapping a tagged corpus through combination of existing heterogeneous taggers. In: Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC) (2002)
6. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
7. Boser, B., Guyon, I., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, pp. 144–152. ACM Press, New York (1992)
8. Oostdijk, N., Goedertier, W., van Eynde, F., Boves, L., Martens, J.P., Moortgat, M., Baayen, H.: Experiences from the spoken dutch corpus project. In: Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC), pp. 340–347 (2002)
9. van Eynde, F.: Part of speech tagging en lemmatisering. Technical report, Centrum voor Computerlinguïstiek, K.U. Leuven (2000)
10. Giménez, J., Márquez, L.: SVMTool: A general POS tagger based on support vector machines. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC) (2004)
11. Nakagawa, T., Kudo, T., Matsumoto, Y.: Unknown word guessing and part-of-speech tagging using support vector machines. In: Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium, pp. 325–331 (2001)
12. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
13. Alpaydin, E.: Introduction to Machine Learning. MIT Press, Cambridge (2004)

Towards Adaptive Web Mining: Histograms and Contexts in Text Data Clustering

Krzysztof Ciesielski and Mieczysław A. Kłopotek

Institute of Computer Science, Polish Academy of Sciences,
ul. Ordona 21, 01-237 Warszawa, Poland
{kciesiel,klopotek}@ipipan.waw.pl

Abstract. We present a novel approach to the growing neural gas (GNG) based clustering of the high-dimensional text data. We enhance our *Contextual* GNG models (proposed previously to shift the majority of calculations to context-sensitive, local sub-graphs and local sub-spaces and so to reduce computational complexity) by developing a new, histogram-based method for incremental model adaptation and evaluation of its stability.

1 Introduction

New visual Web clustering techniques (like WebSOM of Kohonen et al . [11]) target at creating multidimensional document maps in which geometrical vicinity would reflect conceptual closeness of documents in a given document set. Various extensions to this approach to handle document organization under non-stationary environment conditions of growing document collections, like adaptive hierarchical document organization supported by human-created concept-organization [8], attempts to capture the move of topics, dynamic enlargements of document maps [5,13]. All these approaches concentrate on pure introduction of the organization of documents in space spanned by the term dimensions.

In our research project BEATCA [10], we proposed a novel approach, addressing both the issue of topic drift and scalability. In brief, we introduced the concept of so-called contextual map. While being based on a hierarchical (three-level) approach, it is characterized by four distinctive features. To overcome SOM rigidity, we propose first to use modified growing neural gas (GNG [6]) clustering technique. The GNG is then projected onto a 2D map, which is less time consuming than direct WebSOM like map creation. Any other structural clustering (e.g. artificial immune network approach) can be used instead[1]. The second innovation is the way we construct the hierarchy: we split the documents into (many) independent clusters (we call "contexts"), then apply structural clustering within them, and in the end cluster structurally the "contexts" themselves. Drastic dimensionality reduction within the independent clusters is gained (as they are more uniform topically) which accelerates the process and stabilizes it.

[1] Contextual aiNet model has also been implemented in BEATCA and proved to be an efficient alternative for "flat" models [2].

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 284–295, 2007.

The third innovation is the way we apply GNG technique. Instead of the classical global search, we invented a mixed global/local search especially suitable for GNG [10]. And the forth point is that we turned to analyzing the structure of the term dimensions themselves. Roughly speaking, we believe that the profile of a document cluster should not be characterized alone by the average term frequency (or more generally "importance"), but rather by the term (importance) distribution. We approximate these distributions by respective histograms. As it turn out, such an approach allows for much better fitting of new documents into clusters, improves clustering performance with respect to external quality measures (e.g. class purity) and hence can be used as an improved cluster quality measure itself, as the experimental section will demonstrate.

1.1 Quality Measures for the Contextual Clustering

Clustering is a learning process with hidden learning criterion, intended to reflect some esthetic preferences, like: uniform split into groups (topological continuity) or appropriate split of documents with known a priori categorization. A number of clustering quality measures have been developed in the literature [15,7], checking how the clustering fits the (hidden) expectations.

However, those widely-used supervised and unsupervised measures are not sufficient for complete evaluation and comparison of meta-clustering graph-based models, such as GNG graphs, SOM maps or AIS idiotypic networks. Their major disadvantages include:

- instead of subgraph or map area, they take into account individual units (nodes in GNG, cells in SOM, antibodies in AIS), which do not represent real data cluster; they disregard similarity relationship between units and their relative locations
- they do not allow fair comparison of models consisting of different number of units, built in subspaces of different dimensionality (such as individual contextual models)

First of the two above mentioned problems can be solved by introducing measures evaluating correctness of the model structure. We proposed such measures in [2] and [3]. The latter problem can be tackled by measures based on histograms of distributions the function pondering the terms in different models, different subspaces and/or different cells. We discuss such approach in this paper.

1.2 Paper Organization

The rest of this paper is organized as follows. In section 2 a novel, histogram-based vector space representation and its applications are presented. In particular, section 2.4 describes context adaptation procedure based on histograms. Section 3 discusses issues of histogram-based contextual models evaluation and presents experiments on contextual GNG-based model incrementality and stability. Final conclusions from our research work are given in section 4.

2 Histograms in Vector Spaces

As has been said in previous sections, the coordinate value referring to the term t_i in the vector d_j representing the whole document is equal to the value of the pondering (term-weighting) function $f(t_i, d_j)$. This function may ponder the term globally (like TFxIDF, $f_{t_i,d_j} \cdot log\left(\frac{N}{f_{t_i}}\right)$), or locally like the contextual function $f_G(t_i, d_j) = m_{t_i,G} \cdot f_{t_i,d_j} \cdot log\left(\frac{f_G}{f_{t_i}}\right)$, with $m_{t_i,G} = \frac{\sum_{d \in G}\left(f_{t_i,d} \cdot m_{dG}\right)}{f_{t_i,D} \cdot \sum_{d \in G} m_{dG}}$).[2]

In the following subsection we will extend this representation by an information about the structure of dimensions in the vector space. Subsequently we will describe possible applications of this information.

2.1 Distributions of the Function Pondering the Terms

Properties of each term can be considered individually (for a single document), or in the broader context of a given (sub)set of documents D. In the latter case we can consider the values of the pondering function for a given term t for each document $d \in D$ as observed values of a random variable with underlying continuous probability distribution. In practical cases the continuous distribution will be approximated by a discrete one, so that the information about the random variable distribution for the term t can be summarized as a histogram $H_{t,D}$.

Let us consider the document $d \in D$ and the pondering function f. We shall represent the document d by a normalized vector $d = \left[f'_{t_0,d}, \ldots, f'_{t_T,d}\right]$, where $f'_{t_i,d} = \|d\|^{-1} \cdot f_{t_i,d}$ for $i = 0, \ldots, T$. After normalization, all the documents are located within the unit hypercube $[0,1]^T$.

For a fixed number $Q_{t,D}$ of intervals of the histogram $H_{t,D}$ we define the discretization $\Delta_{t,D} : [0,1] \mapsto \{0, \ldots, Q_{t,D} - 1\}$, i.e. the transformation of the normalized pondering function into the index of interval.

In the simplest case it can be a uniform split of the interval $[0,1]$ into segments of equal length $\Delta_{t,D}\left(f'_{t,d}\right) = \lfloor (Q_{t,D} - 1) \cdot f'_{t,d} \rfloor$. An efficient discretization, however, should take into account the fact, that the pondering function for a fixed term takes values in only a subset of the unit interval (like in the case of splitting the set of documents into homogenous subsets, as done in contextual approach). Optimal discretization should also approximate quantile-based split of the underlying pondering function distribution.

The next important issue here is the choice of the optimal number of intervals for the histogram. A too large number of intervals has a few negative implications, including an excessive memory consumption for storage of histograms and deterioration of approximation of the continuous probability distribution of the

[2] $f_{t,d}$ is the number of occurrences of term t in document d, $f_{t,D}$ is the total frequency of term t in all documents from collection D, m_{dG} is the degree of document d membership level in group G (after fuzzy-clustering), m_{tG} is the degree of membership of term t in group G, f_G is the number of documents in group G, f_t is the number of documents containing term t, N is the total number of documents.

values of the pondering function (compare experiments in section 3.2). This is especially true for small subsets of the document collection where a single interval may represent just a couple of documents, which implies a high variance of frequencies in neighboring intervals. One could in this case of course use regularization (smoothening) of histograms, e.g. via weighed averaging, but this would mean not only an additional computational burden (with incremental learning multiple repetitions would be necessary), but also, as experiments show, the histogram based classification results would not be improved.

On the other hand, an underestimation of the number of intervals also negatively influences the quality of histogram based reasoning. With the decrease of the number of intervals, the information contained in them gets closer to the pondering function of a given term, and the multidimensional distribution concentrates around the weight center (centroid) of the subspace represented by the set of documents D. In the extreme case of only one interval, the binary information of occurrence or non-occurrence of a term in the set D is represented.

For the discretization $\Delta_{t,D}$ and a fixed interval q_i let us define the characteristic function:

$$\chi\left(\Delta_{t,D}\left(f'_{t,d}\right), q_i\right) = \begin{cases} 1 & \text{if } \Delta_{t,D}\left(f'_{t,d}\right) = q_i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Then we define the value of the interval q_i of the histogram $H_{t,D}$ for the term t in document collection D as:

$$H_{t,D}(q_i) = \sum_{d \in D} \chi\left(\Delta_{t,D}\left(f'_{t,d}\right), q_i\right) \tag{2}$$

So individual intervals of a histogram H_{tD} represent the number of occurrences of a discretized value of the pondering function f for the term t in the document collection D. The interval values can be in turn transformed to relative frequencies via the plain normalization $H'_{t,D}(q_i) = T_{t,D}^{-1} \cdot H_{t,D}(q_i)$, where $T_{t,D} = \sum_{i \in \Theta} H_{t,D}(q_i)$ is the total number of documents $d \in D$ containing the term t, $\Theta = \{0, \ldots, Q_{t,D} - 1\}$. The frequency distribution approximates the probability distribution of the unknown variable describing the weight of occurrence of term t in randomly chosen document $d \in D$. A term not occurring in any document of the collection will be represented by an "empty" histogram, in which all intervals and the corresponding probabilities will have value equal zero.

2.2 Significance of a Term in a Context

With the exponential increase of dictionary size for a document collection, the most important task is the identification of the most significant terms, most strongly influencing clustering and classification of documents as well as the description of the resulting clusters (keywords extraction). Also the impact of non-significant terms (the number of which grows much more rapidly than the number of significant ones) needs to be bounded.

The first stage in differentiating the term significance is the dictionary reduction process. It is kind of "binary" differentiation: non-significant terms are omitted from further stages of document processing. The dictionary reduction can be conducted in two phases: the global one and the contextual one.

Beside dictionary reduction (removal of least important terms), introduction of contextual pondering function [10] leads also to diversification of the significance of the remaining terms. We are interested also in similar diversification expressed as a function of features of term histograms. Intuitively, less significant terms are represented by histograms with the following features:

- high value of curtosis (a histogram with high peaks), which is especially visible for terms that are frequent and occur uniformly in the document collection, hence are less characteristic
- the domain (the carrier) of the histogram is relatively "compact", having few intervals with non-zero coordinates meaning low variation of pondering function values
- non-zero values occur only for intervals with low indices (corresponding to low values of pondering function)

Dually, the significant terms are those that are not too common, have highly differentiated values of pondering function (many non-zero intervals), and at the same time the pondering function values are high (non-zero values of intervals with high indices).

Therefore we define the significance of a term t in the document collection D as follows:

$$w_{t,D} = \frac{\sum_{i \in \Theta} ((i+1) \cdot H_{t,D}(q_i))}{Q_{t,D} \cdot T_{t,D}} \tag{3}$$

where $\Theta = \{0, \ldots, Q_{t,D} - 1\}$, $H_{t,D}(q_i)$ is the value of the interval q_i of the histogram $H_{t,D}$, and $T_{t,D}$ is the sum of values of all intervals (i.e. the total number of occurrences of term t in D). It is normalized: $w_{t,D} \in [0, 1]$.

The above measure has the advantage that it can be computed in a fixed time (if the sums from the nominator and denominator are stored separately), and can be updated at low computational cost when documents appear and disappear in a given subspace or context (see also section 2.4).

2.3 Determining the Degree of Membership of a Document to a Context

A document fits well to a given contextual subspace if the distribution of some measurable features of term occurrence is typical for the "majority" of documents in this space. Generally, we can look here at features like correlations or co-occurrences of some terms or location-based statistics (e.g. deviance of distances between repeated occurrences of a term in the document content from the Poisson distribution).

Qualitative features can also be taken into account, like style characteristics (dominant usage of a synonym or non-typical inflection) or even features not directly related to the textual content (e.g. link structure between hypertext

documents). But in this paper we restrict ourselves to the analysis of frequency of term occurrence and to a definition of "typicality" based on histograms of pondering function for individual terms in a given context. Hence we can talk about an approach similar to statistical maximum likelihood estimation, in which, based on observed values, we construct a parametric function approximating an unknown conditional probability distribution $f \propto P(D|\Theta)$. The likelihood function should maximize the probability of observed data, and on the other hand it should highly valuate unseen data similar to ones in the training sample (the "generalization" capability of the model). We proceed the same way in our case. A document is considered as "typical" for which the values of the pondering function for the *majority* of terms are frequent ones in the given context.

Additionally, to avoid domination of the aggregated term-based function evaluating document "typicality" by less important (but more numerous) terms, the aggregation should take into account the formerly defined term significance in a given context. Therefore, the similarity (degree of membership) of the document d to the context determined by the document collection D is defined as follows:

$$m_f(d', D) = \frac{\sum_{t \in d'} w_{t,D} \cdot H'_{t,D}(q_i)}{\sum_{t \in d'} w_{t,D}} \tag{4}$$

where $w_{t,D}$ is the significance of a term (eq. (3)), $H'_{t,D}$ is the normalized histogram for the term t (see section 2.1), and $q_i = \Delta_{t,D}(f_{t,d'})$ is the sequential index of the interval, determined for a fixed normalized pondering function f, context D, and discretization $\Delta_{t,D}$. The function $m_f(d', D)$ takes its values in $[0, 1]$.

It should be noted that the cost of computing the similarity function $m_f(d', D)$ is $O(|d'|)$, and it is proportional to the number of distinct terms in the document and equal to the complexity of the cosine measure calculation.

Having determined the similarity of a document to individual contexts in the contextual model, we obtain the vector of fuzzy memberships of a document to the contexts, similarly to known methods of fuzzy clustering (e.g. Fuzzy-ISODATA). In the next section we explain, how such a vector is used to achieve incremental updates of the contextual GNG model.

2.4 Incremental Adaptation of Contexts

While the topic distribution within the stream of documents is dynamically changing in time (e.g. some Internet or intranet documents appear, disappear or have its content modified) also the contextual clustering models have to be adapted correspondingly. Such adaptation is performed both on the level of individual documents and the document clusters, represented by GNG cells. So a new document can be assigned to a context, and within it to a GNG cell. A modified document may be moved from one GNG cell to another GNG cell, in the same, or in another context. As a result, cells may not fit their original GNGs and it may be necessary to move them elsewhere, as a side effect of the so-called reclassification.

When a single new document is appearing, its similarity to every existing context is calculated by equation (4) and the document is assigned to its "most

similar" context[3]. Whenever document context is modified, it may eventually be removed from its previous context and assigned to a new one.

Important aspect of context adaptation is that contextual term importance measure (eq.(3)) can be efficiently updated as documents are added or removed from a given context. Constant-time update of the importance of each term t which appears in document D requires only to keep separately numerator and denumerator from equation (3) and to update them adequately. Denominator is increased or decreased by one, while nominator by $i+1$, where i is the index of the updated interval in the histogram $H_{t,D}(q_i)$. Index i is computed by the discretization $\Delta_{t,D}\left(f_{t,d'}\right)$ (conf. section 2.1).

After any of the contextual GNG models has converged to a stable state, the reclassification procedure is applied. Each document group is represented by reference vector within a cell, which can be treated as a pseudo-document d_{v_i}. The similarity of d_{v_i} to every other (temporally fixed) context is calculated (eq.(4)). If the "most similar" context is different from the current context then the cell (with assigned documents) is relocated to corresponding contextual model. The relocated cell is connected to the most similar cell in the new GNG graph.

There is no room to go into details, so we only mention that also the whole context can be eventually incorporated within some other context, on the basis of our between-context similarity measure, based on Hellinger divergence. Finally, we obtain incremental text data meta-clustering model, based both on adaptive properties of modified GNG model (within-context adaptation, [3]) and on dynamically modified contexts, which allows for clustering scalability and adaptation also on inter-context level.

3 Experimental Results

To evaluate the quality of incremental GNG-based document processing, we compared it to the global and hierarchical approaches. The architecture of BEATCA system supports comparative studies of clustering methods at any stage of the process (i.e. initial document grouping, initial topic identification, incremental clustering, model projection and visualization, identification of topical areas on the map and its labeling [10]). In particular, we conducted series of experiments to compare the quality of GNG, AIS and SOM models for various model initialization methods, winning cell search methods and learning parameters [10,2]. We have also investigated properties of the contextual representation and its impact on clustering model convergence [3].

In [3] we argued that the standard clustering quality measures are not sufficient to evaluate all aspects of meta-clustering model such as GNG. We proposed graph-structure evaluation measure based on comparison of GNG structure with minimal spanning tree structure. In the next section, we define histogram-based reclassification procedure, which facilitates evaluation of the contextual model stability. In section 3.2 we present experimental results on 20 Newsgroups and

[3] One could also consider assignment of a single document to more than one context (fuzzy assignment).

large sample of ca.100.000 html pages to evaluate impact of contextual approach on incremental model stability. The scalability study in section 3.3 was based on a collection of more than one million Internet documents, crawled by BEATCA topic-sensitive crawler .

3.1 Histogram-Based Reclassification Measures

Each contextual model (as well as subgraph or map area) represents some topically consistent (meta-)cluster of documents. Traditionally, such a cluster is represented by a single element (e.g. centroid, medoid, reference vector in GNG/SOM, antibody in immune model). Alternative representation of a group of documents have been presented in section 2. It has numerous advantages such as abandoning of the assumption of spherical shape of clusters and efficient adaptation during incremental learning on dynamically modified data sets. It also allows for the construction of various measures for subspace clusters evaluation. Here we focus only on one such measure, evaluating reclassification properties of contextual groups.

Reclassification aims at measuring stability of the existing structure of the clustering model (both on the meta-level of contexts and on the level of document groups in some subgraphs and map areas). Reclassification measures also the consistency of the histogram-based subspace description with the model-based clustering. For the fixed clustering structure (e.g. some split of the document collection into contexts) we can describe each cluster by a set of histograms, like in section 2.1. Having such histograms built, we can classify each document to its "most similar" histogram-based space, like in section 2.3. Finally, we can investigate the level of agreement between original (model-based) and the new (histogram-based) grouping.

In the ideal case we expect both groupings to be equal. To assess how far from ideal agreement two groupings are, we construct contingency table. Since the group indexes in original and the new grouping are left unchanged, correctly reclassified objects appear in the diagonal elements of the contingency matrix. Finally, we can calculate measures traditionally used for evaluation of classifiers performance (precision, recall, F-statistics, etc.).

3.2 Contextual Clustering Reclassification Quality

In this section we present short summary of contextual clustering stability evaluation based on histogram reclassification measure (section 3.1). Two series of experiments were executed on 20 Newsgroups collection and a sample of 96805 web pages crawled from Polish Internet. We compared the influence of vector space representation (contextual vs TFxIDF), representation dimensionality reduction (number of distinct terms in dictionary) on reclassification results.

In case of contextual representation, the reclassification quality for 20 Newsgroups was 19997 of 20000 correctly reclassified documents (almost 100%). Most of the 119 errors were due to the documents that didn't contain any significant terms after dictionary dimensionality reduction (so they were assigned to

a *"null"* context). What is an interesting result is that reclassification based on less reduced dictionary (15000 terms left, versus 7363 in former case) led to significantly worse results (90% correctly reclassified documents). Again, it supports our claim that the abundance of low-quality terms (exponentially growing) can overshadow inner clustering structure of documents collection. On the other hand, introduction of term-importance factor and contextual representation facilitate reclassification. In case of standard TFxIDF weights only 18567 (93%) document assignments were correct.

For the collection of 96805 Polish Web pages results were also good. HTML documents were grouped into 265 contexts of diverse size (from a few to more than 2500 pages in a single group), so the reclassification task seemed to be rather tricky. Still, for contextual weights 93848 (97%) pages were reclassified correctly, versus 89974 (93%) for TFxIDF weights. Dictionary consisted of 42710 terms, selected among more than 1.2 million.

To sum up, we investigated the impact of the number of histogram intervals on reclassification. Obviously, too low number of intervals had negative impact (17374 correct answers for 3-interval histograms). However, almost any reasonably chosen number of intervals led to high reclassification quality (19992 for 6, 19997 for 10, 19991 for 20, 19956 for 50 and 19867 correct answers for 100 interval histograms). The lower is the number of documents in the collection and the lower diversity of contexts is, the less intervals in required. Too large number of intervals can distort histogram shape regularity and aggravate approximation of the pondering function distribution, since there is not enough documents used to approximate each interval value.

3.3 Scalability Issues

To evaluate scalability of the proposed contextual approach (both in terms of space and time complexity), we built a model for a collection of more than one million documents crawled by our topic-sensitive crawler, starting from several Internet news sites (cnn, reuters, bbc).

Resulting model consisted of 412 contextual maps, which means that the average density of a single map was about 2500 documents. Experimental results in this section are presented in series of box-and-whisker plots, which allows to present a distribution of a given evaluation measure (e.g. time, model smoothness or quantization error) over all 412 models, measured after each iteration of the learning process (horizontal axis). Horizontal lines represent median values, area inside the box represents 25% - 75% quantiles, whiskers represent extreme values and each dot represents outlier values.

Starting with the initial document clustering/context initialization via hierarchical Fuzzy ISODATA, followed by GNG model learning and GNG-to-SOM projection (model visualization), the whole cycle of map creation process took 2 days. It is impressing result, taking into account that Kohonen and his coworkers reported processing times in order of weeks [12]. It should also be noted that the model was built on a single personal computer. As it has been stated

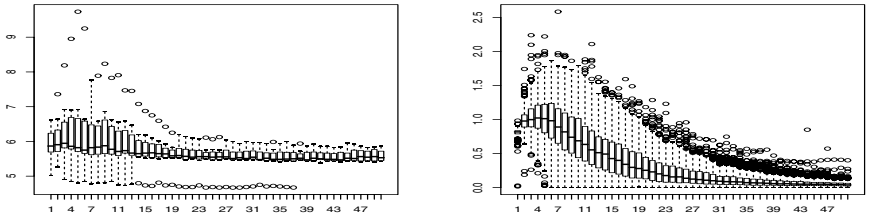


Fig. 1. Contextual model computation complexity (a) execution time of a single iteration (b) average path length of a document

before, contextual model construction can be easily distributed and parallelized, what would lead to even shorter execution times.

The first observation is the complexity of a single iteration of the GNG model learning (Figure 1(a)), which is almost constant, regardless of the increasing size of the model graph. It confirms the observations from section 3, concerning efficiency of the tree-based winner search methods. One can also observe the positive impact of homogeneity of the distribution of term frequencies in documents grouped to a single map cell. Such homogeneity is - to some extent - acquired by initial split of a document collection into contexts. Another cause of the processing time reduction is the contextual reduction of vector representation dimensionality.

In the Figure 1(b), the dynamic of the learning process is presented. The average path length of a document is the number of shifts over graph edges when documents is moved to a new, optimal location. It can be seen that model stabilizes quite fast; actually, most models converged to the final state in less than 30 iterations. The fast convergence is mainly due to topical initialization. It should be stressed here that the proper topical initialization can be obtained for well-defined topics, which is the case in contextual maps.

The Figure 2 presents the quality of the contextual models. The final values of average document quantization (Figure 2(a)) and the map quantization (Figure 2(b)) are low, which means that the resulting maps are both "smooth" in terms of local similarity of adjacent cells and precisely represent documents grouped

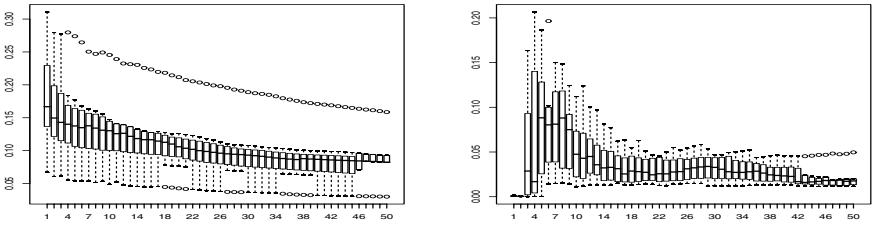


Fig. 2. Contextual model quality (a) Average Document Quantization (b) Average Map Quantization

in a single node. Moreover, such low values of document quantization measure have been obtained for moderate size of GNG models (majority of the models consisted of only 20-25 nodes - due to their fast convergence - and represented about 2500 documents each).

4 Concluding Remarks

We presented a new concept of document cluster characterization via term (importance) distribution histograms. This idea allows the clustering process to have a deeper insight into the role played by the term in formation of a particular cluster. So a full profit can be taken from our earlier idea of "contextual clustering", that is of representing different document clusters in different subspaces of a global vector space. We have also elaborated incremental methods of document cluster models based both on GNG model properties and histogram-based context adaptation. Such an approach to mining high dimensional datasets proved to be an effective solution to the problem of massive data clustering. The contextual approach appears to be fast, of good quality and scalable (with the data size and dimension). Additionally, the histogram-based characterization of document clusters proved to be a stabilizing factor in creating the clustering structure, and well suited for document classification. As a side effect, a new internal cluster quality measure, based on histograms, has been developed.

We believe that the idea of histogram-based subspace identification and evaluation can be efficiently applied not only to textual, but also other challenging high dimensional datasets (especially those characterized by attributes from heterogeneous or correlated distributions).

Acknowledgements

This research has been partially supported by the European Commission and by the Swiss Federal Office for Education and Science with the 6th Framework Programme project no. 506779 "Reasoning on the Web with Rules and Semantics" (REWERSE, cf. http://rewerse.net). Krzysztof Ciesielski was partially supported by MNiSW grant no. N516 005 31/0646.

References

1. Bezdek, J.C., Pal, S.K.: Fuzzy Models for Pattern Recognition: Methods that Search for Structures in Data. IEEE, New York (1992)
2. Ciesielski, K., Wierzchon, S., Klopotek, M.: An Immune Network for Contextual Text Data Clustering. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 432–445. Springer, Heidelberg (2006)
3. Ciesielski, K., Klopotek, M.: Text Data Clustering by Contextual Graphs. In: Todorovski, L., Lavrač, N., Jantke, K.P. (eds.) DS 2006. LNCS (LNAI), vol. 4265, pp. 65–76. Springer, Heidelberg (2006)

4. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science 41, 391–407 (1990), citeseer.nj.nec.com/deerwester90indexing.html
5. Dittenbach, M., Rauber, A., Merkl, D.: Uncovering hierarchical structure in data using the Growing Hierarchical Self-Organizing Map. Neurocomputing 48(1-4), 199–216 (2002)
6. 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, vol. 7, pp. 625–632. MIT Press, Cambridge, MA (1995)
7. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On clustering validation techniques. Journal of Intelligent Information Systems 17(2-3), 107–145 (2001)
8. Hung, C., Wermter, S.: A constructive and hierarchical self-organising model in a non-stationary environment. In: International Joint Conference in Neural Networks (2005)
9. Klopotek, M.: A new Bayesian tree learning method with reduced time and space complexity. Fundamenta Informaticae 49(4), 349–367 (2002)
10. Klopotek, M., Wierzchon, S., Ciesielski, K., Draminski, M., Czerski, D.: Techniques and Technologies Behind Maps of Internet and Intranet Document Collections. In: Lu, J., Ruan, D., Zhang, G. (eds.) E-Service Intelligence – Methodologies, Technologies and Applications. Springer-Verlag Series: Studies in Computational Intelligence, vol. 37, X., p. 711, Springer, Heidelberg (2007)
11. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 30. Springer, Heidelberg (2001)
12. Kohonen, T., Kaski, S., Somervuo, P., Lagus, K., Oja, M., Paatero, V.: Self-organization of very large document collections, Helsinki University of Technology technical report (2003), http://www.cis.hut.fi/research/reports/biennial02-03
13. Rauber, A.: Cluster Visualization in Unsupervised Neural Networks, Diplomarbeit, Technische Universität Wien, Austria (1996)
14. Timmis, J.: aiVIS: Artificial Immune Network Visualization. In: Proceedings of EuroGraphics UK 2001 Conference, Univeristy College London, pp. 61–69 (2001)
15. Zhao, Y., Karypis, G.: Criterion functions for document clustering: Experiments and analysis, available at http://www-users.cs.umn.edu/~karypis/publications/ir.html

Does SVM Really Scale Up to Large Bag of Words Feature Spaces?

Fabrice Colas[1], Pavel Paclík[2], Joost N. Kok[1], and Pavel Brazdil[3]

[1] LIACS, Leiden University, The Netherlands
{fcolas,joost}@liacs.nl
[2] ICT Group, Delft University of Technology, The Netherlands
p.paclik@ewi.tudelft.nl
[3] LIACC-NIAAD, University of Porto, Portugal
pbrazdil@liacc.up.pt

Abstract. We are concerned with the problem of learning classification rules in text categorization where many authors presented Support Vector Machines (SVM) as leading classification method. Number of studies, however, repeatedly pointed out that in some situations SVM is outperformed by simpler methods such as naive Bayes or nearest-neighbor rule. In this paper, we aim at developing better understanding of SVM behaviour in typical text categorization problems represented by sparse *bag of words* feature spaces. We study in details the performance and the number of support vectors when varying the training set size, the number of features and, unlike existing studies, also SVM free parameter C, which is the Lagrange multipliers upper bound in SVM dual. We show that SVM solutions with small C are high performers. However, most training documents are then bounded support vectors sharing a same weight C. Thus, SVM reduce to a nearest mean classifier; this raises an interesting question on SVM merits in sparse *bag of words* feature spaces. Additionally, SVM suffer from performance deterioration for particular training set size/number of features combinations.

1 Introduction

Classifying text documents into categories is difficult because the size of the feature space is very high, commonly exceeding tens of thousands. The *bag of words* feature space, where each dimension corresponds to the number of occurrences of the words in a document, is most often used in text classification because of its wide use in information retrieval and its implementation simplicity. The number of training documents, which is several orders of magnitude smaller than the feature space size is another difficulty. The most prominent technique applied to text classification is *linear* Support Vector Machines. First introduced to text categorization by [Joachims, 1998], SVM were systematically included in subsequent comparative studies [Dumais et al., 1998], [Yang and Liu, 1999], [Zhang and Oles, 2001] and [Yang, 2003]. Their conclusions suggest that SVM is an outstanding method for text classification. The paper [Forman, 2003] also confirms SVM as outperforming other techniques.

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 296–307, 2007.

However, in several large studies, SVM did not systematically outperform other classifiers. For example [Davidov et al., 2004], [Colas and Brazdil, 2006], [Schönhofen and Benczúr, 2006] showed that depending on experimental conditions, kNN or naive Bayes can achieve better performance. In [Liu et al., 2005], the difficulty to extend SVM to large scale taxonomies was also shown. In this regard, experimental set-up parameters can have a more important effect on performance than the individual choice of a particular learning technique [Daelemans et al., 2003]. Indeed, classification tasks are often highly unbalanced; the way training documents are sampled has a large impact on performance. To do fair comparisons between the classifiers, the aggregating multi-class strategy which generalizes binary classifiers to the multi-class problem, e.g. SVM one-versus-all, should be taken into account. Better results are achieved if classifiers natively able to handle multi-class are reduced to a set of binary problems and then, aggregated by pairwise classification [Fürnkranz, 2002].

For these reasons, we only study situations where the number of documents in each class is the same and we choose binary classification tasks as the baseline of this work. Selecting the right parameters of SVM, e.g. the upper bound C, the kernel, the tolerance of the optimizer ϵ and the right implementation are also non-trivial. In text categorization, *linear* kernel is commonly regarded as yielding to the same performance than non linear kernels [Yang and Liu, 1999]. Our previous study [Colas and Brazdil, 2006] suggested that large optimizer tolerances ϵ achieved similar performance than smaller ones, whereas training cost reduced largely as ϵ was set large. A *linear* kernel and a large tolerance are used in this work. We investigate the effect of C, seldomed optimized in text categorization, when the number of training documents and the size of the *bag of words* are varying. Effect of C is observed in terms of performance and through the nature of the SVM solutions.

In our extensive empirical study, SVM solutions with small C are high performers but when C nears zero, most training documents are bounded support vectors with equal weight C. Then, SVM reduces to the *nearest mean classifier* and optimisation in SVM is superfluous. Consequently, we raised the question whether SVM really scale-up to large *bag of words* feature spaces. In addition, SVM suffer from performance deterioration for particular training set size/number of features combinations. It was already described in our previous study and to our knowledge, SVM was not noted before [Colas and Brazdil, 2006]. Finally, the side effects of training SVM in a discrete space like the *bag of words* feature space are discussed.

The outline of this paper is the following. We recall some aspects of SVM, we describe our methodology and we detail our experimental results. Then, we consider some related work before doing the concluding remarks.

2 Background on Support Vector Machines and Text Classification

Support Vector Machines are based on statistical learning theory [Vapnik, 1995]. Its theoretical foundations together with the results obtained in various fields makes it a popular algorithm in machine learning.

In text categorization, the vector space is referred to as the *bag of words* feature space where each dimension corresponds to the number of occurrences of words in a document. So let the vector space be $\mathbf{N}^d$ where d is the dimension. The training examples are denoted by $\mathbf{x}_i \in \mathbf{N}^d$ with $i = 1, ..., N$, N being the training set size. We consider the task of classifying unseen points $\mathbf{x}$ into two classes $\{-1; +1\}$. SVM classification function is achieved by $\Phi_{SVM}(\mathbf{x}) = sign(\langle \mathbf{w}.\mathbf{x} \rangle + b)$ where $\mathbf{w}$ are the coordinates of the separating hyperplane and the scalar b is the plane bias to origin. The hyperplane $\mathbf{w}$ is the one which separates with maximum distance points of the two classes. This concept is referred to as the *geometrical margin* $\gamma = 1/2||\mathbf{w}||_2$ and its maximum is searched. Maximizing the margin γ is equivalent to minimizing $||\mathbf{w}||_2/2$. In the case where the classes are separable, derivation of SVM hyperplane is formalized by the following optimization problem:

$$\begin{aligned} &minimize_{\mathbf{w},b} && \tfrac{1}{2}\langle \mathbf{w}.\mathbf{w} \rangle \\ &subject\ to && y_i(\langle \mathbf{w}.\mathbf{x}_i \rangle + b) \geqslant 1 \end{aligned} \tag{1}$$

This primal form depends on the size of the input feature space. The dual form is obtained by posing the Lagrangian and deriving it according to $\mathbf{w}$ and b:

$$\begin{aligned} &maximize_{\alpha} \textstyle\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i.\mathbf{x}_j \rangle \\ &subject\ to \quad \textstyle\sum_{i=1}^{N} y_i \alpha_i = 0 \\ &\qquad\qquad 0 \leqslant \alpha_i \leqslant C, i = 1...N \end{aligned} \tag{2}$$

The dual form depends on the number of documents and not anymore on the feature space. It is a desirable property in text classification because the number of features of the *bag of words* feature spaces is usually very large. In order to limit values of Lagrange coefficients α_i, an upper bound C is introduced so that each training document has a maximum contribution when classes are not linearly separable. In particular, as $\mathbf{w} = \sum_{i=1}^{N} y_i \alpha_i \mathbf{x}_i$ and $0 \leqslant \alpha_i < C$:

$$\begin{aligned} &\lim_{C \to 0} ||\mathbf{w}||_2 = 0, \\ &\lim_{C \to 0} \gamma = \infty \end{aligned} \tag{3}$$

SVM solution interpretation First, we recall that a set of points is *convex* if the line segment between any two of its points stays within the set [Strang, 1986]. In addition, we consider the smallest of those convex sets and we further refer to it as the smallest convex hull. Thus, the training of SVM on a two classes problem reduces to the identification of the points that lie on the smallest convex hull of each two classes, i.e. their boundary. In the case where classes are linearly separable, the solution of linear SVM is the hyperplane in force equilibrium between the smallest convex hulls of those two classes.

The force pressure exerted by the points on the hyperplane position is quantified by the Lagrange multipliers (α_i). Thus, the points that are within the smallest convex hull of their respective classes are set inactive with $\alpha_i = 0$. On the other hand, active points are those lying on the boundary of each class. Those points are referred to as support vectors (SV) and they are defined as any point $\mathbf{x}_i$ for which $\alpha_i > 0$. The higher is α_i, the more it contributes to the positioning of the hyperplane.

However, when classes are not linearly separable, points may exert high pressure on the hyperplane without ever being on the right side of it. Consequently, some multipliers may be very large compared to others or even infinite. In order to limit the individual contribution of the multipliers, the so-called soft margin is introduced. In a soft margin SVM solution, the multiplier values are upper bounded by a parameter C, that is the maximal cost that we are ready to pay to classify a training point well. There are four types of training points; we list and characterize them by their distance and their contribution to the hyperplane in the following table.

Type	Distance	Contribution	State	Classification
(1)	$y_i(\langle \mathbf{w}.\mathbf{x}_i \rangle + b) \geqslant 1$	$\alpha_i = 0$	inactive	well classified
(2)	$y_i(\langle \mathbf{w}.\mathbf{x}_i \rangle + b) = 1$	$0 < \alpha_i < C$	active, *in* bound	well classified
(3)	$0 < y_i(\langle \mathbf{w}.\mathbf{x}_i \rangle + b) < 1$	$\alpha_i = C$	active, *at* bound	well classified
(4)	$y_i(\langle \mathbf{w}.\mathbf{x}_i \rangle + b) < 0$	$\alpha_i = C$	active, *at* bound	misclassified

The concept of sparsity aims at finding the most parsimonious representation for a problem. In a sparse SVM solution, most of the training points are set inactive (1). The ones that are active (2,3,4) lie on the smallest convex hull of each two classes. They are expected to represent the two classes concepts. In linearly separable problems, there are only training points of types (1) and (2) (without the bound C). However, most of the problems are not linearly separable, which means that the linear separation surface will misclassify part of the training points. Thus, in soft margin SVM, the more a solution has of bounded SV (3 and 4), the less linearly separable the problem is. In addition, we remark that only the bounded SV of type (4) are misclassified training points, in comparison with the bounded SV of type (3) that are well classified. Large proportion of bounded SV is not desirable because it illustrates the non linear separability of the problem. However, using non-linear kernels may not show any improvement if, e.g. training points of distinct classes overlap at the same location in the feature space. No surface of any complexity can separate well those overlapping training points.

Settings of text classification. In addition to large number of features, the *bag of words* feature space exhibits high levels of sparsity. The majority of the word occurrences are zero. As the dimensionality of the problem increases, there will be more training points on the smallest convex hull of the two classes. As an example, more than 75% of the dual variables are non-zero in the SVM solutions of [Rousu et al., 2006] in text classification. We also illustrate this phenomenon through our experiments.

The concept of force equilibrium between the two classes is formalized by the constraint: $\sum_{y_i \in \{+1\}} \alpha_i = \sum_{y_j \in \{-1\}} \alpha_j$, where the sum of the individual training point forces should remain equal for the two classes. We consider a specific SVM solution where all the documents are equally weighted and we further refer to it as the *nearest mean classifier* solution. Our experiments suggests that the C parameters yielding to the best performing SVM solutions in high feature spaces, produce SVM solutions that are similar to the *nearest mean classifier*. In that situation, most of the training point share the same weight.

3 Methodology

To investigate whether SVM scale up to large *bag of words* feature spaces, we study the nature and the performance of SVM when the number of features increases and when documents are added to the training set. We characterize SVM solutions by the Lagrange multiplier values; the number of SV *within* and *at* bound are recorded. We study the effect of the C parameter on the solutions and we identify the values which yield the best performance. The kernel is not considered as a parameter in this work because previous studies in text categorization suggest that *linear* kernel performs similarly as more complex ones [Yang and Liu, 1999].

Dimensions of experimentation. We systematically balance the number of training documents in each class. With respect to the feature space, words are ranked and then selected by information gain at the start of each new experiment. We do measurements in coordinates of a two dimensional space determined by series of exponential values on each axis; $2^{\frac{i}{2}+b}$ with $i = 1, 2, 3...$ and $b \in \mathbf{N}$; when $b = 6$ the series start with $\{90, 128, 181, 256, ...\}$. Values of C are chosen similarly; 10^i $\{0.0001, 0.001...1000\}$.

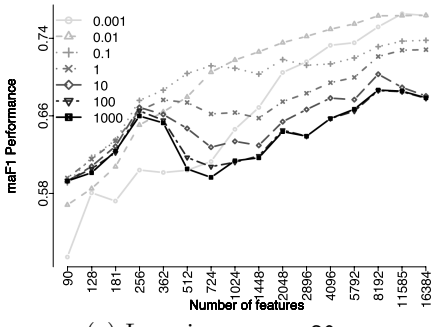
Evaluation methodology. As in [Yang and Liu, 1999], we adopt the macro averaged F_1 measure $maF_1 = \frac{2 \times maPrecision \times maRecall}{maPrecision + maRecall}$ which relates precision and recall computed in two confusion matrices, interchanging the definition of target class in the two-class problem. Measure statistics are estimated by 10-fold cross validation; mean of maF_1 and mean of the number of SV *at* and *within* bounds.

Corpora. We select two binary text classification problems from our previous study involving more than 200 tasks [Colas and Brazdil, 2006]. The first task is `Bacterial Infections and Mycoses` against `Disorders of Environmental Origin` or `C01-C21` for short, from `Ohsumed-all` data set which is a corpora of medical abstracts classified into 23 cardio vascular disease categories. The second task is `alt.atheism` against `talk.religion.misc` or `20ng` for short, from `20newsgroups` which is a data set containing 20000 emails taken from 20 usenet newsgroups. The `libbow` of [McCallum, 1996] is used to do the preprocessing; no stemming is performed.

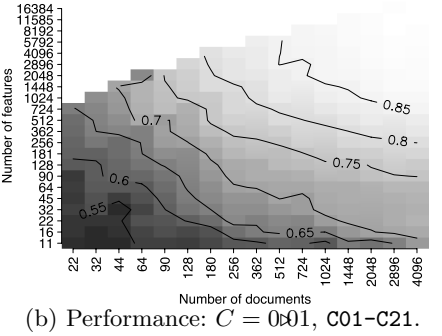
SVM implementation. We use the `libsvm` package [Chang and Lin, 2001]. The ϵ parameter controling the tolerance of the stopping criterion is set to 0.1 although other smaller tolerances were investigated. While no effect on the performance was observed, this setting significantly reduced the training time. As we based our experiments on the `libbow` package in our previous study [Colas and Brazdil, 2006], by reproducing the learning curves with `libsvm` we discard implementation as a source of variability for our conclusions.

4 Experimental Results and Discussion

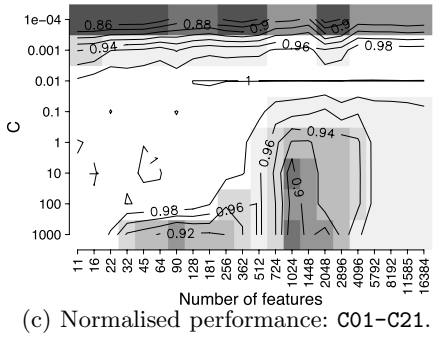
Figure 1 illustrates the performance behaviour of SVM solutions when experimental settings are varying. The number of training (resp. test) documents in (a) is set to 1448 (resp. 200) and to 4096 (resp. 547) in (c). In (a), SVM performance is illustrated by classical learning curves when the size of the feature



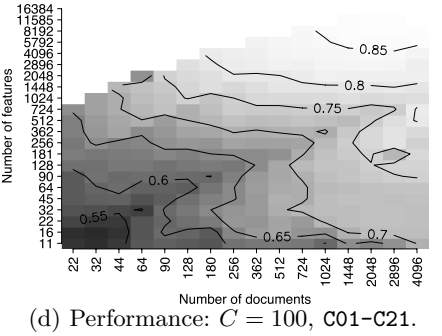
(a) Learning curves: `20ng`.



(b) Performance: $C = 0.01$, `C01-C21`.



(c) Normalised performance: `C01-C21`.



(d) Performance: $C = 100$, `C01-C21`.

Fig. 1. Figure (a), (b) and (d) illustrate that SVM performance generally increases as the feature space and the training set are increasing. Depending on the C, performance may show a dip as in (a) around 724 features (`20ng`) and in (c) around 1448 features (`C01-C21`). In addition, (d) illustrates the linear dependency of the dip to the number of training documents and of features. Finally, (c) shows that the range of best performing C (in white, contour line 1) reduces towards small C values as the feature space is increasing.

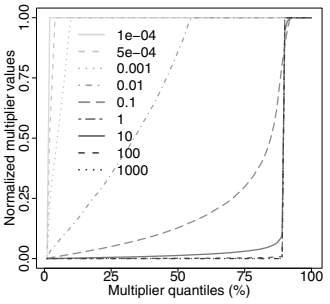
space is increasing and for different values of C; the problem is `20ng`. Similarly, the performance pattern of SVM on `C01-C21` is illustrated in (c) but this time, in order to discard the performance variability associated to the *bag of word* size, performance is normalised relative to the one of the best C setting for each number of features. Thus, the contour line labelled 0.94 should be understood as 6% below the performance of the best performing C setting ($C = 0.01$), whose contour line is itself 1. In (b) and (d), the performance differences between solutions with small (0.01) and large (100) C is illustrated through the contour lines, when the number of features and the size of the training set are varying.

Best performing SVM Figure 1 (a) and (c) illustrate the effect of C on the performance as the number of features is varying. In small feature spaces, i.e. 90-256 in (a) and 11-362 in (c), most C settings perform similarly. However, as the feature space is further increased, performance varies widely from one setting to another; for several C settings a performance dip occurs. For larger feature spaces, very small C values like $\{0.01, 0.001\}$ are best performing. The relative difference to the best C setting may be as small as 2-4%, between contour lines 0.96 and 0.98 in (c), but it depends on the classification problem. In (c), the setting $C = 10^{-4}$ yields systematically lower performance, because all training documents have then equal weight, i.e. 100% bounded SV. This will be discussed in a later paragraph.

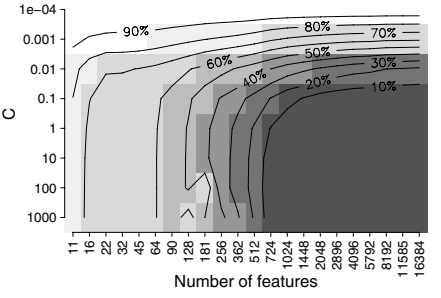
Through (a) and (c), C is shown as a parameter to tune to obtain the best of SVM but most of previous studies use default values of the implementation[1]. If C is tuned, the best performances are achieved for the largest feature spaces, thus making feature selection unnecessary. Finally, (b) and (d) show as expected, that more training documents yield better performance.

Figure 2 illustrates the nature of SVM solutions for different experimental settings. The number of training (resp. test) documents in (a), (b) and (c) is set to 4096 (resp. 547) on `C01-C21`. In (a), the multiplier values α_i of the ten SVM solutions of the cross validation are normalised by C. Then, they are ordered along the x-axis by increasing value. The quantiles are in x and the normalised multiplier values (α_i/C) in y. In (a), the solutions for different C are described when there are 11585 features, because through Figure 1 we show that feature selection is unnecessary for SVM. In (b), the proportion of bounded SV is illustrated when the size of the feature space is varying and for different C values. The contour lines characterize the proportion of bounded SV. Finally, (c) presents the variation of the mean of the total number of SV as estimated by the 10-fold cross validation, when the feature space size is varying and $C = 100$. The curve of the total number of SV is decomposed through the number of bounded and unbounded SV. In particular, we detail SVM solution characteristics for $C = 100$ in (c) because for such large C, SVM performance displays a dip in Figure 1,

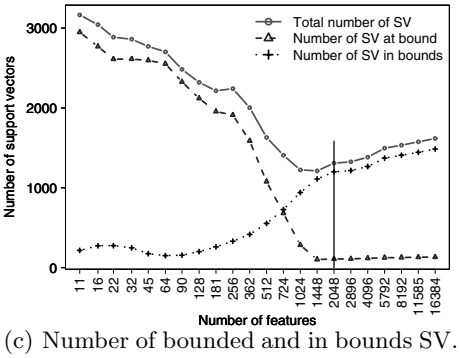
[1] As mentionned by a reviewer, SVMLight sets the default value of C according to a heuristic. Most other SVM implementations have fixed C default values (e.g. WEKA, libbow and libsvm).



(a) Quantiles of α_i/C.



(b) Proportion of bounded SV.



(c) Number of bounded and in bounds SV.

Fig. 2. In (a), the proportion of bounded SV ($\alpha_i/C = 1$) increases as C diminishes, and there are only bounded SV for very small $C \in \{10^{-4}, 5 \times 10^{-4}\}$. In (b), the proportion of bounded SV is high for all C (11-362 features), but it decreases for higher feature space sizes as shown in (c). First, the total of SV follows the decrease in bounded SV (11-362). Then, the decrease is partially compensated by the appearance of unbounded SV (362-16384). The total of SV exhibits a dip slightly before 2048 features. In (b), for very small $C = 10^{-4}$, 100% of the SV are bounded.

when the number of features matches the number of training documents per class, i.e. 2048 for a training set of 4096 documents.

Nature of SVM solutions We relate Figure 1 (c) with Figure 2 (b) first. For small feature spaces (11-362 features), any C setting was previously shown to perform equally well. In the shade of Figure 2 (b), we now characterize those solutions as having large proportions of bounded SV with 30 to 100% of the training points that are bounded SV. As a consequence, those training points also share equal weight ($\alpha_i = C$). If only the proportions between inactive training points, bounded and unbounded SV are considered, then those solutions are similar. This explains partly why the performance of the soft margin solutions for a large range of C settings yields the same performance, i.e. most of the training points are equally weighted.

Besides, training points are located in a countable number of positions because the *bag of word* feature space is discrete and sparse. With few selected features representing the problem, training points of distinct classes will overlap. Thus, no hyperplane of any complexity can separate those points, they are unseparable. Their multiplier values tend to infinity; soft margin SVM force them to $\alpha_i = C$ and most SV are bounded. At constant feature space size, when the number of training documents increases, the overlap effect will accentuate.

In contrary to solutions in small feature spaces where there are mainly bounded SV, SVM solutions in large feature spaces have mostly unbounded SV as illustrated in Figure 2 (c). Furthermore, the number of unbounded SV keeps raising as the feature space increases. This illustrates that every point tends to define its own, local, class boundary. Thus, every training point tends to locate on the smallest convex hull of its class in large feature spaces.

In Figure 2 (c), the number of bounded SV stagnates for any large feature space. These bounded SV are outliers, i.e. wrongly labelled training points that lie within the other class concept. Thus, no linear separation classify well those points. In Figure 1, a possible origin of the performance dip for high C might relate to those outliers, whose weight is to important compared to regular points.

On Figure 2 (b), when $C = 10^{-4}$, 100% of the training points are bounded SV. Selecting a particular number of features, e.g. 11585 features in Figure 2 (a) or any other large feature space (362-16384) in (b), there tend to be 100% of bounded SV when C lowers. Given the limit value of the *geometrical margin*, which tends to infinity as C tends to zero, all training points become active and fall within the *geometrical margin* bounds when C lowers. In that situation, SVM act as the simple *nearest mean classifier* because all training points are equally weighted ($\alpha_i = C$), irrespective of their good classification or not.

A performance drop for SVM In Figure 1 (a), we observe a performance dip for 724 features and $C \in \{1, 10, 100, 1000\}$. In Figure 1 (c), which plots normalised SVM performance in the second classification task, the dip also occurs around (1024-2048) features when $C \in \{1, 10, 100, 1000\}$. Therefore, the dip occurs when the number of features matches the number of training documents per class, i.e. 724 for `20ng` in Figure 1 (a) and 2048 for `C01-C21` in Figure 1 (c). On Figure 1 (d) where $C = 100$, the performance dip is illustrated by a line that passes through the origin and that shades off the line. Finally, Figure 2 (a) in association with Figure 1 (c) when $C = 100$, illustrate that the performance dip occurs as the total number of SV exhibits a drop in the same settings.

In our previous study [Colas and Brazdil, 2006], this performance drop appeared on a large number of classification problems. Therefore, the drop is not specific to a classification problem, although its emphase does. It also does not relate to the choice of the SVM implementation because analogous behaviour is observed for both `libbow`, that was used in [Colas and Brazdil, 2006] and `libsvm` that we adopt in this study. We attribute the phenomenon to the decrease in bounded SV not matching the increase of *within* bounds SV as the number of features increases. However, most of the training points become bounded SV as

C lowers so that there is no transfer between bounded and unbounded SV. Thus, because most of the training points are already bounded SV, learning curves for small C are unaffected by the performance dip.

5 Related Work

In [Cristianini and Shawe-Taylor, 2000], it is mentioned that *although the maximal margin classifier does not attempt to control the number of support vectors, [...] in practice there are frequently very few support vectors.* However, in this empirical study, we illustrate that SVM solution are not sparse in large feature spaces. A similar observation in text classifiction was made in the recent work of [Rousu et al., 2006], who observed more than 75% of the dual variables that were non-zero, i.e. $\alpha_i > 0$. In [Burges and Crisp, 1999] and [Rifkin et al., 1999], SVM solution properties are analysed in terms of uniqueness and of degeneracy. In particular, the conditions for which all dual variables are at bound, refered to as degenerate solutions, are described. In our experiments, we also revealed experimental settings for which every training point is bounded SV and we explained that they yielded trivial SVM solutions, i.e. *nearest mean classifier*. Furthermore, the study of [Mladenic et al., 2004] raises the question whether sparsity of the feature space is a more reliable parameter in predicting classifier performance than the number of features. Our experiments confirmed the specificity of the *bag of words* and its discrete nature.

6 Conclusion

Based on large-scale experimentation, we systematically described the nature of SVM solutions in text classification problems when the number of training documents, the number of features and the SVM constraint parameter (C) are varying. In order to study SVM performance in equal grounds to other classification methods (e.g. kNN and naive Bayes in our previous study), training data are systematically balanced and only binary classification tasks were considered.

Usually, SVM are expected to produce *sparse solutions* that should present good generalization properties. Moreover, SVM execution time, i.e. test time, depends on solution sparsity, which makes sparsity a desirable property in real applications with many documents to classify. However, our empirical study shows that tightly constrained SVM solutions with small C are high performers. When C lowers, most of the training documents tend to become bounded support vectors. As every training document has then equal weight, this raises a question on true merits of the SVM classifier for large *bag of word* feature spaces.

In fact, when all the training documents have equal weight, SVM solution acts as the *nearest mean classifier*. Expressed differently, the *nearest mean classifier* is an SVM solution where the individual contribution of all training documents are set equal ($\alpha_i = C$). In our experiments, we come to that situation when C nears zero. In addition, training points that fall within the *geometrical margin* of the hyperplane are bounded. Thus, as C tends to zero, all training documents

become bounded SV because the *geometrical margin* tends to infinity. Letting C nears zero will, in practice, generate unwanted trivial SVM solutions.

Finally, our empirical study reveals a performance dip of SVM when the number of features matches approximately the number of training documents per class. As in other statistical classifiers, the dip may illustrate a *peaking phenomenon*. To investigate this issue, additional experiments will be carried out that record both the performance in the training and the test set, to eventually detect situation leading to overfitting. However, SVM is well known to control overfitting. Our alternative hypothesis is therefore to relate directly the dip to the discrete nature of the *bag of word*, and not SVM itself. We plan to investigate the dip and to find ways of improving SVM performance by further characterizing the nature of SVM solutions in text categorization. In reference to the no free lunch theorem, we do not expect SVM to outperform systematically other techniques. However, a more complete understanding of the *bag of words* specificities and of SVM solutions should reduce the number of problems where SVM underperform other methods.

Acknowledgements. The research of Fabrice Colas and Joost Kok is supported by the Netherlands Bioinformatics Centre (NBIC) through its research programme BioRange. The research of Pavel Paclik is supported by the applied science division STW of NWO under the project DTF.5699. The authors wish to express their gratitude to David Tax for clarifying the link between the *geometrical margin* and C.

References

[Burges and Crisp, 1999] Burges, C., Crisp, D.: Uniqueness of the svm solution. In: Proceedings of the 12th Conference on Neural Information Processing Systems, MIT Press, Cambridge, MA (1999)

[Chang and Lin, 2001] Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001)

[Colas and Brazdil, 2006] Colas, F., Brazdil, P.: On the behavior of svm and some older algorithms in binary text classification tasks. In: Proceedings of TSD2006, Text Speech and Dialogue, pp. 45–52 (2006)

[Cristianini and Shawe-Taylor, 2000] Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and other kernel based learning methods. Cambridge University Press, Cambridge (2000)

[Daelemans et al., 2003] Daelemans, W., Hoste, V., De Meulder, F., Naudts, B.: Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, Springer, Heidelberg (2003)

[Davidov et al., 2004] Davidov, D., Gabrilovich, E., Markovitch, S.: Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. In: Proceedings of the 27th annual international conference on Research and development in information retrieval, pp. 250–257 (2004)

[Dumais et al., 1998] Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization. In: CIKM '98: Proceedings of the seventh international conference on Information and knowledge management, pp. 148–155. ACM Press, New York, US (1998)

[Forman, 2003] Forman, G.: An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research 3, 1289–1305 (2003)

[Fürnkranz, 2002] Fürnkranz, J.: Pairwise classification as an ensemble technique. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS (LNAI), vol. 2430, pp. 97–110. Springer, London, UK (2002)

[Joachims, 1998] Joachims, T.: Making large-scale support vector machine learning practical. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods: Support Vector Machines, MIT Press, Cambridge, MA (1998)

[Liu et al., 2005] Liu, T.-Y., Yang, Y., Wan, H., Zeng, H.-J., Chen, Z., Ma, W.-Y.: Support vector machines classification with a very large-scale taxonomy. SIGKDD Exploration Newsletter 7(1), 36–43 (2005)

[McCallum, 1996] McCallum, A. K.: Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering (1996), http://www.cs.cmu.edu/~mccallum/bow

[Mladenic et al., 2004] Mladenic, D., Brank, J., Grobelnik, M., Milic-Frayling, N.: Feature selection using linear classifier weights: interaction with classification models. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 234–241. ACM Press, New York, US (2004)

[Rifkin et al., 1999] Rifkin, R., Pontil, M., Verri, A.: A note on support vector machine degeneracy. In: Algorithmic Learning Theory, 10th International Conference, ALT '99, Tokyo, Japan, December 1999, vol. 1720, pp. 252–263. Springer, Heidelberg (1999)

[Rousu et al., 2006] Rousu, J., Saunders, C., Szedmak, S., Shawe-Taylor, J.: Learning hierarchical multi-category text classification models. Journal of Machine Learning Research (2006)

[Schönhofen and Benczúr, 2006] Schönhofen, P., Benczúr, A.A.: Exploiting extremely rare features in text categorization. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 759–766. Springer, Heidelberg (2006)

[Strang, 1986] Strang, G.: Introduction to applied mathematics. Wellesley-Cambridge Press (1986)

[Vapnik, 1995] Vapnik, V.N.: The Nature of Statistical Theory. Information Science and Statistics. Springer, Heidelberg (1995)

[Yang, 2003] Yang, Y.: A scalability analysis of classifiers in text categorization. In: SIGIR-03, 26th ACM International Conference on Research and Development in Information Retrieval, ACM Press, New York, US (2003)

[Yang and Liu, 1999] Yang, Y., Liu, X.: A re-examination of text categorization methods. In: SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, pp. 42–49. ACM Press, New York (1999)

[Zhang and Oles, 2001] Zhang, T., Oles, F.J.: Text categorization based on regularized linear classification methods. Inf. Retr. 4(1), 5–31 (2001)

Noise Filtering and Microarray Image Reconstruction Via Chained Fouriers

Karl Fraser[1], Zidong Wang[1], Yongmin Li[1], Paul Kellam[2], and Xiaohui Liu[1]

[1] School of Information Systems, Computing and Mathematics
Brunel University, Uxbridge, Middlesex, UB8 3PH, U.K.
karl.fraser@brunel.ac.uk
[2] Department of Infection
University College London, London, W1T 4JF, U.K.

Abstract. Microarrays allow biologists to determine the gene expressions for tens of thousands of genes simultaneously, however due to biological processes, the resulting microarray slides are permeated with noise. During quantification of the gene expressions, there is a need to remove a gene's noise or background for purposes of precision. This paper presents a novel technique for such a background removal process. The technique uses a gene's neighbour regions as representative background pixels and reconstructs the gene region itself such that the region resembles the local background. With use of this new background image, the gene expressions can be calculated more accurately. Experiments are carried out to test the technique against a mainstream and an alternative microarray analysis method. Our process is shown to reduce variability in the final expression results.

Keywords: Microarray, Filtering, Reconstruction, Fourier.

1 Introduction

The invention of the microarray in the mid-90's dramatically changed the landscape of modern day genetics research. The devices allow simultaneous real time monitoring of expression levels for tens of thousands of genes. One of these so-called "gene chips" contains probes for an organism's entire transcriptome. The different conditions or cell lines render a list of genes with their appropriate activation levels. These gene lists are then analysed with the application of various computational techniques, for example clustering [1], or modelling [2] such that differential expressions are translated into a better understanding of the underlying biological phenomena.

A major challenge with any real-world data analysis process is how to address data quality issues effectively. Although microarray hardware is engineered to very high tolerances, noise (henceforth "noise" and "background" are synonymous) will be introduced into the final output slide. This noise can take many forms, ranging from common artefacts such as; hair, dust and scratches on the slide, to technical errors like; the random variation in scanning laser intensity or the miscalculation of gene expression due to alignment issues. Alongside these

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 308–319, 2007.

technical errors there exist a host of biological related artefacts; contamination of complementary Deoxyribonucleic Acid (cDNA) solution or inconsistent hybridisation of the multiple samples for example.

Unfortunately, these images are expensive to produce and the current climate is such that "bad" experiment sets must still be analysed, regardless of their quality. Whereas such poor images could simply be discarded in other fields, here, an image must yield some knowledge irrespective of how small. It is common practice throughout then to implement some form of duplication in-situ such that correction tasks can take place during downstream analysis. Much work in the field therefore focuses on post-processing or analysing the gene expression ratios themselves [1,3,4,5,6,7] as rendered from given image sets, which means there is relatively little work directed at pre-processing or improving the original images to begin with [8,9].

Microarray images are full of background signal that is of no real interest specifically to the experimental process. Nevertheless, these artefacts can have a detrimental effect on the identification of genes as well as their accurate quantification. There are many reasons for this, the most critical of which is due to the similar intensity levels seen between noise and a gene (due to inappropriate DNA binding sites for example). In this paper, we present an algorithm that attempts to remove the biological experiment from the image. In this context, the biological experiment consists of the gene spot regions. Put another way; imagine the image is made up of two separate layers. The bottom layer consists of the glass substrate material upon which the gene spots are deposited onto to begin with. The top layer on the other hand consists of the gene spots. Removal of the biological experiment regions is to clear the top layer such that the hidden regions of the bottom layer can be seen. In effect, this removal process is equivalent to background reconstruction and will therefore produce an image which resembles the "ideal" background more closely in experimental regions. Subtracting this new background from the original image should yield more accurate gene spot regions. The reconstructed expressions are contrasted to those as produced by GenePix [10] (a commercial system commonly used by biologists to analyse images) and O'Neill *et al.* [9] (one of the first reconstruction processes implemented to deal with microarray image data).

The paper is organised in the follow manner. First, we formalise the problem area as it pertains to real microarray image data sets and briefly explain the workings of contemporary approaches in the next section. Section Three discusses the fundamental idea of our approach with the appropriate steps involved in the analysis. In Section Four, we briefly describe the data used throughout the work then detail and evaluate the tests carried out for the synthetic and real-world data. Section Five summarises our findings, draws out some relevant conclusions and defines future considerations and directions.

2 Background

Regardless of the specific techniques used to assist with downstream analysis of microarray image data, all of them have similarities due to the nature of the

problem. For example, the techniques require knowledge of a given gene spot's approximate central pixel as well as the slide's structural layout. A boundary is then defined around the gene spot and background pixels with the median of these regions taken to be foreground and background intensities respectively. Then, the background median is subtracted from the foreground and the result is summarised as a $\log_2$ ratio. Other bounding mechanisms include pixel partitioning via histogram [3,11] and region growing [12,13] functions with a detailed comparison of the more common approaches given in [8]. The underlying assumption for these mechanisms is that there is little variation within the gene and background regions.

Unfortunately, this is not always the case as seen in Fig. 1a generally, which depicts a typical test set slide (enhanced to show gene spot locations) with a total of 9216 gene regions on the surface and measuring ~5000×2000 pixels. A good example of the low-level signal produced in the image can be seen in the close-up sections, where problems such as missing or partial gene spots, shape inconsistencies, and background variation can be seen. Such issues are highlighted in more detail in b and c where the scratch and background illuminations around the presented genes change significantly.

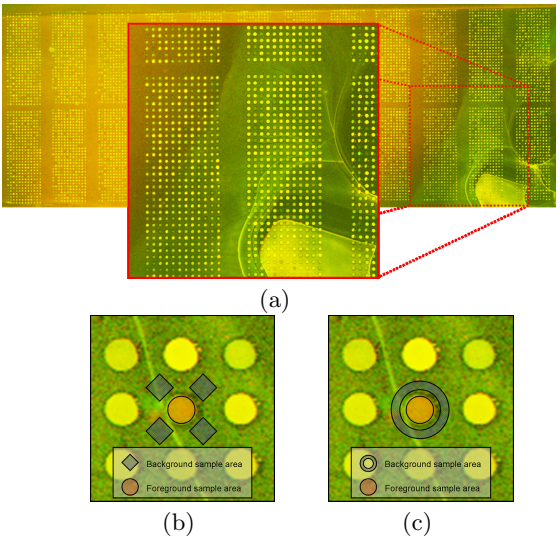


Fig. 1. Example Images: Typical test set Slide Illustrating Structure and Noise (a) and Sample Gene, Background Locations for GenePix Valleys (b) and ImaGene Circles (c)

What is needed is a more specific background determination process that can account for the inherent variation between the gene and background regions. Texture synthesis represents a possible avenue for such background reconstruction processes. An established reconstruction technique is that as proposed by Efros *et al.* [14] whereby a non-parametric process grows a reconstruction outwards

from an initial seed pixel, one pixel at a time via Markov Random Fields. The Bertalmio *et al.* [15] approach on the other hand attempts to mimic techniques as used by professional restorers of paintings and therefore works on the principle of an isotropic diffusion model. Moreover, Chan *et al.* [16] greatly extended the work of [14] and others to propose a curvature model based approach. However, microarray images contain thousands of regions requiring such reconstructions and are therefore computationally expensive to examine with the highlighted techniques. In an attempt to overcome such time restrictions (although not focused at microarray data itself) Oliveira *et al.* [17] aimed to produce similar results to [15] albeit quicker, although as we shall see the approach loses something in translation.

One of the first reconstruction techniques applied specifically to microarray images is that as proposed by O'Neill *et al.* [9] which utilises a simplification of the Efros *et al.* [14] technique. In this context, a gene spot is removed from the surface and recreated by searching a known background region and selecting pixels most similar to the known border. By making the new region most similar to given border intensities it is theorised that local background structures will transition through the new region. However, the best such a process has accomplished in this regard is to maintain a semblance of valid intensities, while the original topological information is lost. The next section describes a technique that attempts to address some of these issues, e.g. retention of topology, process efficiency, and edge definition in a more natural way.

3 A New Analysis Technique

In this work, we have proposed Chained Fourier Image Reconstruction (*CFIR*), a novel technique that removes gene spot regions from a microarray image surface. Although this may seem counter-intuitive (the gene spots are the elements of value in a microarray after all), the successful removal of these regions leads to more accurate or natural looking background surfaces, which can be used to yield yet more accurate gene spot intensities. Techniques such as O'Neill work in the spatial domain exclusively and essentially compare all gene border pixels to those of the local background to produce appropriate pixel mappings. Although this works well, such brute force methods are typically expensive with respect to execution time. However, if we harness the frequency domain along with more traditional spatial ideas we can render a reconstruction that inherently deals with the issues (illumination, shading consistencies etc) more efficiently.

Taking the diagram of Fig. 1b as our reference, let us now detail the CFIR process outlined in Table 1. Initially due to the nature of the microarraying process, gene spots can be rendered with different shapes and dimensions, both individually and through the channels. Therefore, a generic window centred at the gene (as determined by GenePix) can be used to capture all pixels p_{xy} within a specified square distance from this centre, where *(x,y)* are the relative coordinates

of the pixels in the window centred at pixel *p*. Window size is calculated directly from an analysis of the underlying image along with resolution meta-data if needed. The window can then be used to determine the appropriate *srcList* and *trgList* pixel lists (foreground and background) accordingly. Note that in the current implementation, the background region resembles a square as defined by the outer edges of the diamonds in Fig. 1b.

With the two lists (*srcList*, *trgList*) in place a Fast Fourier Transform (FFT) is applied to both lists independently (as highlighted in lines 2∼4). If *f(x,y)* for

Table 1. Pseudo-Code of Chained Fourier Transform Reconstruction Function

Input

srcList: List of gene spot region pixels

trgList: List of sample region pixels

Output

outList: srcList pixels recalibrated into trgList range

Function fftEstimation(srcList,trgList):outList

```
1. For each gene
2.   srcMask = fourier transform srcList members
3.   trgSample = fourier transform trgList members
4.   recon = srcMask * trgSample // to generate initial reconstructed surface
5.   While doneIterate = 0
6.     recon = fourier transform initial reconstructed surface
7.     reconPhase = phase elements of reconstructed surface
8.     minimum recon element = smallest element in the trgSample surface
9.     recon = inverse fourier transform merged recon and reconPhase surfaces
           // such that subtle characteristics are retained
10.    recon elements ≤0 = smallest element in srcMask
11.    recon elements ≥65535 = largest element in srcMask
12.    reset non-gene pixels in recon = trgList
13.    if difference between recon and trgSample ¡tolerance
14.      doneIterate = 1
15.    End If
16.  End While
17.  outList = reconstructed region
18. End For
```

End Function

x;y=0,1,...,M-1;N-1 respectively denote the $M \times N$ image region, the digital FFT for *F(u,v)* can be defined as

$$F(u,v) = \sum_{M-1}^{x=0} \sum_{N-1}^{y=0} f(x,y) e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)} \tag{1}$$

where *(u,v)* represent the frequency coordinates of their spatial *(x,y)* equivalents. Note the inverse transform is computed in much the same way. The real R, imaginary I and phase ϕ components of the resulting FFT spectrum can then be broken up according to

$$|F(u,v)| = \left[R^2(u,v) + I^2(u,v)\right]^{\frac{1}{2}}, \text{ and} \tag{2}$$

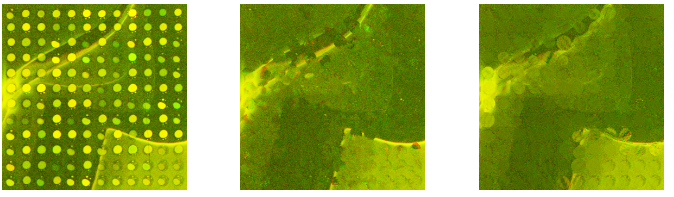
$$\phi(u,v) = tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right], \text{ respectively} \tag{3}$$

Global features of the image regions (repeating patterns, overall region intensity etc) thus become localised within the frequency spectrum, while non-repeating structures become scattered. Retaining this phase information in the reconstructed region is crucial as this has the effect of aligning global features (much the same as the isotropic diffusion approach of [15] does for example) and as such presents subtle surface characteristics (illumination and shading features etc). In order to capture this subtle intensity information within the background (*trgList*) region and allow the gene spot (*srcList*) area to inherit it, a simple minimisation function is used (as per lines 7∼9). More complicated criteria could be computed in this regard but after critical testing it was found that the minimum of the region produces good results and is thus used at present

$$R(u,v) = minimum\,|srcList(R), trgList(R)| \tag{4}$$

The final stage of the algorithm (lines 10∼12) replaces modified background (*trgList*) pixels within the gene spot (*srcList*) area with their original values. Recall, the FFT function disregards spatial information, which means subsequent modifications (like the minimiser function) could well change inappropriate pixels with respect to the reversed FFT of line 9. Therefore, the original non-gene spot pixels must be copied back into the modified regions such that erroneous allocations are not propagated through the reconstructed region during the next cycle.

The actual convergent criterion as highlighted in line 13 can be determined by the mean squared error (MSE) or other such correlation methods between the reconstructed and background regions. Generally though, the MSE falls rapidly over the initial iterations and thenceforth slows until a minimum is reached. Conversely, the correlation coefficient approach would be expected to rise rapidly over the initial iterations and then slow as convergence approaches. Regardless however, the tolerance criterion guarantees termination of the reconstruction process when iterative changes are at a minimum. In practice, tolerance is calculated such that



(a) (b) (c)

Fig. 2. Chained Fourier Transform Example: Original Image (a), Reconstructed O'Neill (b) and CFIR (c) Regions

the absolute difference (for the gene spot (*srcList*) pixels specifically) between all original and reconstructed pairs (for an individual region) are monotonically decreasing. Such monotonicity helps with retention of illumination and tone information that would otherwise be lost. Fig. 2 presents a sample-reconstructed region from the Fig. 1a image as processed by the techniques.

Application of frequency and spatial methods when applied separately to such problems can work well (see Fig. 2b for example) but there are better ways to carry out such processes. The formulation as described for CFIR allows us to inherently combine advantages from both the frequency and spatial domains such that reconstructed regions not only retain implicit domain information but, are processed faster than contemporary methods. Related to this implicit domain information (and suggested in Fig. 2bc) is the problem of correct edge classification. Generally however, CFIR improves handling and production of results accordingly as will be seen in the next section.

4 Experiments

This section details the results of numerous experiments that were designed to test empirically the performance and differences between the O'Neill and CFIR algorithms with respect to GenePix. Although there are many ways that such performance characteristics can be distilled, for this work the focus is on the resulting median expression intensities. These intensities become the raw gene expressions (as used in post-analysis [1,3,4,5,6,7] work for example) and therefore render overall insight into the reconstruction event. In addition, these values allow us to drill down into a particular gene spots repeat set and as such help clarify reconstruction quality.

As it is not possible to determine the optimal background for a gene spot region, the best validation in this context would seem to be to compare against rebuilt background regions. Such a comparison renders a clearer understanding of the reconstruction characteristics. To aid in this, 64 synthetic gene spots (SGS) were created and placed into existing background regions of the Fig. 1a image. If the reconstruction processes were to perform in an ideal manner, the SGSs would be removed from the slide such that these SGS regions are indistinguishable from their local surrounding region.

With the potential errors inherent in the GenePix background generation thus highlighted, the next stage of testing involved determining how these errors translate onto the reconstruction of real gene spots. Experiment two therefore examines the median intensities of the control gene spots for all blocks across the Fig. 1a image and the entire test set. Finally, experiment three carries out an explicit comparison between the techniques and thus yields the gained improvements (or not) of a particular technique as compared with GenePix.

Note that all of the images used in this paper were derived from two experiments conducted using the human gen1 clone set data. These two experiments were designed to contrast the effects of two cancer inhibiting drugs (PolyIC and LPS) onto two different cell lines, one being normal (control, or untreated) and the other the treatment (HeLa) line over a series of several time points. In total, there are 47 distinct slides with the corresponding GenePix results present.

The first experiment is designed to determine how well the reconstruction process can remove a synthetic gene from the image. When removed, the new region can be compared to the original with the difference calculated explicitly. Fig. 3 distils this difference information into a clear plot by calculating the average absolute pixel error for the SGS regions, as determined by the reconstruction techniques.

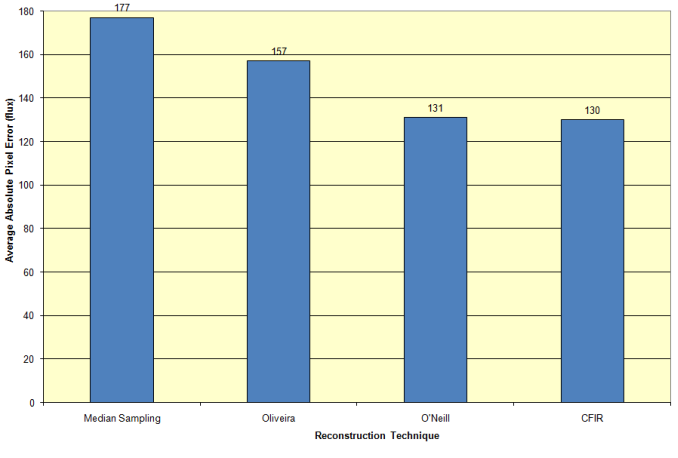
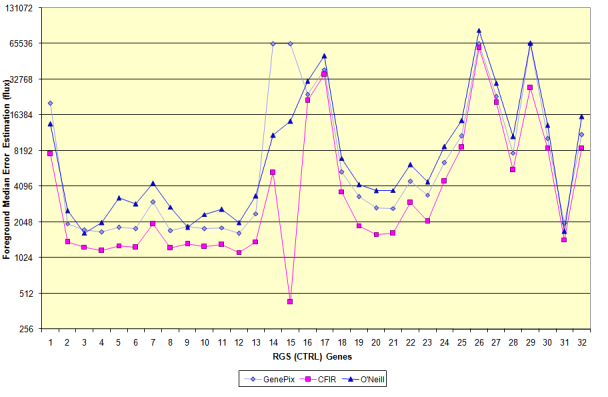


Fig. 3. Synthetic Gene Spots: Average Absolute Pixel Error

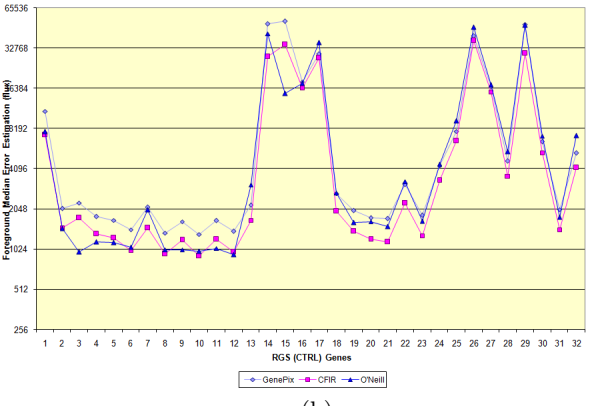
In effect, the graph shows that on average, GenePix's median sampling approach to background classification yields a potential intensity error of 177 flux per pixel for an SGS region, while the other techniques yield smaller error estimates. A consequence of this is that downstream analysis based on GenePix results directly must produce more erroneous gene expressions than realised.

The second experiment (with results as shown in Fig. 4) conducts the performance evaluation of CFIR, O'Neill and GenePix using true gene spots. This

experiment is particularly focused on the relationships between expression measurements for all control repeat genes in the same slide (Fig. 4a) and across all slides in the test set (Fig. 4b). In all cases the underlying assumption for repeated genes is that they (the genes) should have highly similar intensity values (ideally) for a given time point, regardless of their location on the slide surface. Although we would perhaps expect to see some differences in the values as the time points increase over the duration of the biological experiments.



(a)



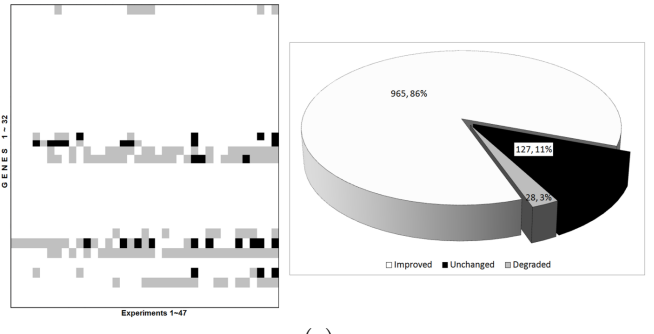
(b)

Fig. 4. Real Gene Spot Curves: Absolute Medians for 32 genes over Fig. 1a (a) and the Test set (b) Regions

The plots represent the absolute foreground median from both image channels for the tested techniques. It is clear from Fig. 4a that CFIR outperformed the other methods comfortably as far as the reduction of individual gene intensities is concerned. However, for the saturated gene spots (gene 15 for example) the estimation difference increases, although it is still closer than GenePix. Generally, for this particular image, CFIR outperformed both O'Neill and GenePix in

relation to reconstruction with the specific residuals at 14514, 7749 and 13468 flux for GenePix, CFIR and O'Neill respectively. Indeed, as far as control data is concerned the CFIR process reduced noise by ∼46%.

Plotting the entire 47-slide test set produces a cross-sectional average through the data. The O'Neill and CFIR processes are much closer overall as compared to the sample slide; however, there are subtle handling differences in the saturated gene regions. Essentially, the intensity jump relationship for the saturated genes has flipped, meaning that CFIR seems to cope with such issues in a smoother way overall. The respective residuals for the entire test set are 10374, 7677 and 9213 flux respectively, which indicates that although not perfect, CFIR tends to produce lower repeat scores in general. To be fair the O'Neill technique has improved the overall score somewhat with respect to the individual image surface. It is clear that reconstructing the true gene spot regions does have a positive



(a)

	Time on Xeon 3.4GHz (hh:mm:ss)		
Image	*GenePix*	*O'Neill*	*CFIR*
Hela PolyIC t00 (1h)	4:00:00+	1:53:00	00:52:39
Hela PolyIC t05	4:00:00+	1:58:00	00:58:15
Hela PolyIC t18	4:00:00+	1:52:00	00:54:44

(b)

Fig. 5. Final Results Comparison: Matrix for test set showing difference in repeat expression fluctuations (a); GenePix, CFIR and Both techniques are assigned the colours black, white and grey (∼10% difference) respectively and Sample Timing Chart (b)

effect on the final expression results but, not so obvious, are the ramifications this reconstruction is having over the entire test set. Fig. 5a therefore plots a comparison chart which shows explicitly the improvement (or not) of a particular reconstruction method as compared to the original GenePix expressions. In addition, execution time plays a critical role in the reconstruction task, as techniques need to run as fast as possible given the number of gene spots that must be processed. Therefore, Fig. 5b presents a brief breakdown of the timings required for the techniques to parse a small percentage of the entire test set.

The distinct banding occurring in gene regions 3∼8 and 16∼19 of Fig. 5a are associated with saturated (or near background) intensities as created by

the scanner hardware and suggest more work is needed with respect to these genes. The non-banded genes on the other hand are indicative of the individual reconstruction techniques being able to account more appropriately for gene intensity replacement. Table 5b highlights the significant speed increase gained by applying the CFIR process to reconstruction rather than the O'Neill and GenePix methods.

5 Conclusions

The paper looked at the effects of applying both existing and new texture synthesis inspired reconstruction techniques to real-world microarray image data. It has been shown that the use of existing methods (which have typically focused on aesthetic reconstructions) to medical image applications can be highly effective, however their output quality and processing time's need to be significantly improved. As for microarray-focused reconstruction, pixel fill order as applied by the O'Neill technique plays a crucial role and should therefore be crafted with greater care.

To overcome timing and accuracy issues, we proposed a novel approach to reconstructing a gene's background by attempting to harness an image's global information more intently along with the gene's neighbour pixels. The proposed technique takes advantage of the grouping concept of the frequency domain and characterises global entities. At the same time, we use local spatial knowledge of a gene to help restrict a constructed regions spread. Results obtained from several experiments showed great improvement over a commonly used package (GenePix) and a brute force approach (O'Neill). Specifically, not only was the gene repeat variance reduced from slides in the test set, but in addition the construction time was decreased $\sim$50% in comparison to O'Neill's technique.

In future studies we wish to investigate gene spots that straddle strong artefact edges along with general transition issues as they are subject to the sub-allocation of replacement pixel(s). A transition edge will have a sharp yet convoluted evolution that can influence the accuracy of a gene's background. Such inconsistencies render themselves in the surface as halos and it would be beneficial if such halos were removed more appropriately. The artefact sub-allocation problem on the other hand requires a subtle approach to correction and we believe a weighted transition map would be more appropriate than the current bivalence approach.

Acknowledgment

This work is in part supported by EPSRC grant (EP/C524586/1) and an International Joint Project sponsored by the Royal Society of the U.K. and the National Natural Science Foundation of China. The authors would also like to thank the anonymous reviewers for their comments.

References

1. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. In: Proceedings of the National Academy of Sciences, USA, December 1998, pp. 14863–14868 (1998)
2. Kellam, P., Liu, X., Martin, N., Orengo, C.A., Swift, S., Tucker, A.: A framework for modelling virus gene expression data. Journal of Intelligent Data Analysis 6(3), 265–280 (2002)
3. Chen, Y., Dougherty, E.R., Bittner, M.L.: Ratio-based decisions and the quantitative analysis of cdna microarray images. Journal of Biomedical Optics 2, 364–374 (1997)
4. Gasch, A.P., Spellman, P.T., Kao, C.M., Carmel-Harel, O., Eisen, M.B., Storz, G., Botstein, D., Brown, P.O.: Genomic expression program in the response of yeast cells to environmental changes. Molecular biology of the cell 11, 4241–4257 (2000)
5. Quackenbush, J.: Computational analysis of microarray analysis. Nature Reviews Genetics 2(6), 418–427 (2001)
6. Kepler, B.M., Crosby, L., Morgan, T.K.: Normalization and analysis of dna microarray data by self-consistency and local regression. Genome Biology 3(7) (2002)
7. Quackenbush, J.: Microarray data normalization and transformation. Nature Genetics 32, 490–495 (2002)
8. Yang, Y.H., Buckley, M.J., Dudoit, S., Speed, T.P.: Comparison of methods for image analysis on cdna microarray data. Journal of Computational and Graphical Statistics 11, 108–136 (2002)
9. O'Neill, P., Magoulas, G.D., Liu, X.: Improved processing of microarray data using image reconstruction techniques. IEEE Transactions on Nanobioscience 2(4) (2003)
10. Anonymous: GenePix Pro Array analysis software. Axon Instruments Inc.
11. Anonymous: QuantArray analysis software. GSI Lumonics
12. Adams, R., Bischof, L.: Seeded region growing. IEEE Transactions on Pattern Analysis and Machine Intelligence 16, 641–647 (1994)
13. Ahuja, N., Rosenfeld, A., Haralick, R.M.: Neighbour gray levels as features in pixel classification. Pattern Recognition 12, 251–260 (1980)
14. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: IEEE International Conference on Computer Vision, pp. 1033–1038. IEEE Computer Society Press, Los Alamitos (1999)
15. Bertalmio, M., Bertozzi, A., Sapiro, G.: Navier-stokes, fluid dynamics, and image and video inpainting. In: IEEE Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2001)
16. Chan, T., Kang, S., Shen, J.: Euler's elastica and curvature based inpaintings. Journal of Applied Mathematics 63(2), 564–592 (2002)
17. Oliveira, M.M., Bowen, B., McKenna, R., Chang, Y.S.: Fast digital image inpainting. In: Proceedings of the Visualization, Imaging and Image Processing, Marbella, Spain, pp. 261–266 (September 2001)

Motif Discovery Using Multi-Objective Genetic Algorithm in Biosequences

Mehmet Kaya

Data Mining and Bioinformatics Laboratory, Department of Computer Engineering,
Fırat University, 23119, Elazığ, Turkey
kaya@firat.edu.tr

Abstract. We propose an efficient method using multi-objective genetic algorithm (MOGAMOD) to discover optimal motifs in sequential data. The main advantage of our approach is that a large number of tradeoff (i.e., nondominated) motifs can be obtained by a single run with respect to conflicting objectives: similarity, motif length and support maximization. To the best of our knowledge, this is the first effort in this direction. MOGAMOD can be applied to any data set with a sequential character. Furthermore, it allows any choice of similarity measures for finding motifs. By analyzing the obtained optimal motifs, the decision maker can understand the tradeoff between the objectives. We compare MOGAMOD with the three well-known motif discovery methods, AlignACE, MEME and Weeder. Experimental results on real data set extracted from TRANSFAC database demonstrate that the proposed method exhibits good performance over the other methods in terms of runtime, the number of shaded samples and multiple motifs.

Keywords: Motif Discovery, Multi-Objective Genetic Algorithms.

1 Introduction

Motif discovery is one of the fundamental problems that have important applications in locating regulatory sites and drug target identification. Regulatory sites on DNA sequence normally correspond to shared conservative sequence patterns among the regulatory regions of corrected genes [8]. These conserved sequence patterns are called motifs. The actual regulatory DNA sites corresponding to a motif are called instances of the motif. Identifying motifs and corresponding instances is very important, so biologists can investigate the interactions between DNA and proteins, gene regulation, cell development and cell reaction under physiological and pathological conditions. The automatic motif discovery problem is a multiple sequence local alignment problem under the assumption that the motif model gives the optimal score for some appropriate scoring function. Solving this problem is NP-complete theoretically. There are several cases for this problem:

1. the simple sample: each sequence in the data set contains exactly one motif instance
2. the corrupted sample: an instance may not appear in every sequence,

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 320–331, 2007.

3. the invaded sample: more than one instance may exist in some sequences,
4. the multiple patterns: the sequences may contain more than a single common motif.

To handle the motif discovery problem, including one or more of the above cases, various approaches and tools were proposed [22, 1-4, 17, 18, 21, 16]. Recently, genetic algorithms (GA) have been used for discovering motifs in multiple unaligned DNA sequences. Of these, Stine et al., [19] presented a structured GA (st-GA) to evaluate candidate motifs of variable length. Fitness values were assigned as function of high scoring alignment performed with BLAST [20]. Liu et al., [13] developed a program called FMGA for the motif discovery problem, which employed the general GA framework and operators described in SAGA [14]. In their method, each individual represents a candidate motif generated randomly, one motif per sequence. Then, Che et al., [5] proposed a new GA approach called MDGA to efficiently predict the binding sites for homologous genes. The fitness value for an individual is evaluated by summing up the information content for each column in the alignment of its binding site. Congdon et al., [6] developed a GA approach to Motif Inference, called GAMI, to work with divergent species, and possibly long nucleotide sequences. The system design reduces the size of the search space as compared to typical window-location approaches for motif inference. They presented preliminary results on data from the literature and from novel projects. Finally, Paul and Iba [26] presented a GA based method for identification of multiple (l, d) motifs in each of the given sequences. The method can handle longer motifs and can identify multiple positions of motif instances of a consensus motif and can extract weakly conserved regions in the given sequences.

However, all of the above studies employ single-objective to discover motifs, although different methods of fitness calculation are used. Also, while in most of the proposed methods the length of motif to be extracted is given beforehand; only one motif per sequence is assumed in some methods. Whereas, multiple similar motifs may exist in a sequence, and identification of those motifs is equally important to the identification of a single motif per sequence. Moreover, almost all of the methods try to find motifs in all of the given sequences. However, some sequences may not contain any motif instance. If it is assumed that a motif instance should be included in all the target sequences, the similarity value used to compare sequences decrease. In this paper, to address all the problems listed and mentioned above, we propose a multi-objective GA based method for motif discovery. The paper demonstrates advantages of multi-objective approach over single-objective ones to discover motifs efficiently and effectively. For this purpose, we use the following three-objective formulation to find a large number of longer and stronger motifs.

$$Maximize\ Similarity, Maximize\ Motif\ \ Length, Maximize\ Support$$

The meanings of these objectives will be discussed in the objectives subsection in detail. Next, we compare the three-objective formulation with the single-objective approach based on the following weighted scalar objective function:

$$Maximize \quad w_1.Similarity + w_2.Motif\ \ Length + w_3.Support$$

We performed experiments on two real data sets to demonstrate the effectiveness of our method. The experimental results show the superiority of the proposed method, in terms of motif length, strongness and the runtime required to find the motifs, over three well known motif discovery methods, AlignACE, MEME and Weeder.

2 Multi-Objective Optimization

Many real world problems involve multiple measures of performance or objectives, which should be optimized simultaneously. Multi-objective optimization (MOO) functions by seeking to optimize the component of a vector-valued objective function. Unlike single-objective optimization, the solution to a MOO problem is a family of points known as the Pareto-optimal set. A general minimization problem of *M* objectives can be mathematically stated as

Minimize $f(x) = [f_i(x), i = 1, \ldots, M]$
subject to:

$$g_j(x) \le 0 \quad j = 1,2,\ldots,J\ ,$$
$$h_k(h) = 0 \quad k = 1,2,\ldots,K$$

where $f_i(x)$ is the i^{th} objective function, $g_j(x)$ is the j^{th} inequality constraint. The MOO problem then reduces to finding x such that $f(x)$ is optimized. In general, the objectives of the optimization problem are often conflicting. Optimal performance according to one objective, if such an optimum exists, often implies unacceptable low performance in one or more of the other objective dimensions, creating the need for a compromise to be reached. A suitable solution to such problems involving conflicting objectives should offer *acceptable*, though possibly sub-optimal in the single objective sense, performance in all objective dimensions, where acceptable is a problem dependent and ultimately subjective concept. An important concept in MOO is that of domination, where a solution x_i is said to dominate another solution x_j if both the following conditions are true:

- the solution x_i is not worse than x_j in all objectives;
- the solution x_i is strictly better than x_j in at least one objective.

This, in turn, leads to the definition of *Pareto-optimality*, where a decision vector $x_i \in U$, where U stands for the universe, is said to be *Pareto-optimal* if and only if there exists no x_j, $x_j \in U$, such that x_i is dominated by x_j. Solution x_i is said to be *nondominated*. The set of all such nondominated solutions is called the *Pareto-optimal set* or the *nondominated set*. . In general, MOO problems tend to achieve a family of alternatives which must be considered the relevance of each objective relative to the others [27]. Recently, some researchers have studied on different problems by using multi-objective genetic algorithms. We have already participated in some of these efforts with data mining area [9-11].

3 The MOGAMOD Algorithm

We use a well-known high-performance multi-objective genetic algorithm called NSGA II [7] to find a large number of motifs from biosequences with respect to three objectives, which will be discussed in the objectives subsection. The NSGA II algorithm is also employed in the two objectives case. On the other hand, we use a standard single-objective GA with a single elite solution in the case of the single-objective formulation.

3.1 Structure of the Individuals

An individual in MOGAMOD represents the starting locations of a potential motif on all the target sequences. An individual expect for its part showing the motif length is divided into n genes, where each gene corresponds to starting location of a motif, if any, in the corresponding sequence. The genes are positional, i.e., the first gene deals with the first sequence, the second gene deals with the second sequence, and so on. Each i^{th} gene, $i = 1,\ldots,n$, is subdivided into two fields: weight (w_i) and starting location (s_i), as shown in Figure 1.

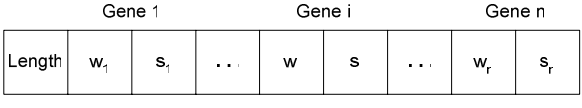


Fig. 1. Representation of an individual

The field weight (w_i) is a real-valued variable taking values in the range $[0..1]$. This variable indicates whether or not the potential motif is present in the corresponding sequence. More precisely, when w_i is smaller than a user-defined threshold (called *Limit*) the motif will not be extracted from the i^{th} sequence. Therefore, the greater the value of the threshold *Limit,* the smaller is the probability that the corresponding sequence will be included in discovering that motif.

The field starting location (s_i) is a variable that indicates the starting location of the motif instance in the i^{th} sequence.

In this study, an extra part in each individual was used to determine the length of the motif. The value of this part changes between 7 and 64 because we restricted the minimum and the maximum length of the predicted motif in those values.

Note that the above encoding is quite flexible with respect to the length of the motifs. A traditional GA is very limited in this aspect, since it can only cope with fixed-length motifs. In our approach, although each individual has a fixed length, the genes are interpreted (based on the value of the weight w_i) in such a way that the individual phenotype (the motif) has a variable length. Hence, different individuals correspond to motifs with different length.

The start of the first population consists of generating, arbitrarily, a fixed number of individuals during the evolution.

3.2 Objectives and Selection

The fitness of an individual in MOGAMOD is assessed on the basis of *similarity*, *motif length* and *support*.

Similarity: It performs a measure of similarity among all motif instances defining an individual. To calculate it, we first generate a position weight matrix from the motif patterns found by MOGAMOD in every sequence. Then, the dominance value (dv) of the dominant nucleotide in each column is found as follows:

$$dv(i) = \max_{b}\{f(b,i)\},\ i = 1,\ldots,l$$

where $f(b,i)$ is the score of the nucleotide b on column i in the position weight matrix, $dv(i)$ is the dominance value of the dominant nucleotide on column i, and l is motif length.

We define the similarity objective function of motif M as the average of the dominance values of all columns in the position weight matrix. In other words,

$$Similarity(M) = \frac{\sum_{i=1}^{l} dv(i)}{l}$$

The closer the value of the similarity (M) to one, the larger the probability that the candidate motif M will be discovered as a motif. The following example shows how to compute the similarity measure in given two position weight matrixes with different size:

Table 1. The position weight matrix of a motif with length 4

	1	2	3	4
A	0.2	1	0	0
C	0.2	0	1	1
T	0.6	0	0	0
G	0	0	0	0

Table 2. The position weight matrix of a motif with length 5

	1	2	3	4	5
A	0.25	0	0.75	1	0
C	0.5	0	0.25	0	0
T	0.25	1	0	0	1
G	0	0	0	0	0

The first matrix (Table 1) implies that the number of target sequences in a dataset may be 5 and the motif length is found to be 4. In such a case, the dominant nucleotide for column 1 is T and its dominance value is 0.6. The dominance values of the other columns are 1. As for the similarity value, it is computed as:

$$\frac{(0.6+1+1+1)}{4} = 0.9$$

Similarly, in the second matrix (Table 2), the number of target sequences may be 8 and the motif length is determined to be 5. The similarity value is computed as 0.85 for this longer motif.

Motif Length: In motif discovery, a motif with large length is always desired because the longer the motif, the lesser is the chance that it simply occurred by chance in the given target sequence.

Support: Here, the meaning of this objective is the same with that in the data mining field. Sometimes, a candidate motif may not appear in every sequence. In other words, one or more sequence may not include any candidate motif. In this case, the aim should become to discover optimal motif, leaving these corrupted sequences out. So, support is the number of sequences composing candidate motifs. The greater the support value, the stronger is the motif covered by most of the sequences in the dataset. Overall, the optimal motif discovery problem is converted into the following three-objective optimization problem:

$$Maximize\ Similarity(M), Maximize\ Motif\ \ Length(M), Maximize\ Support(M)$$

After applying the nondominated sort algorithm, the individuals are assigned a crowding distance and selection is performed using the crowded tournament selection. The crowding distance is computed as the sum of the difference of the objective values of the solutions preceding and following the current solution (corresponding to the individual under consideration) for each objective. This provides a measure of the density of the solution surrounding a particular point in the population.

In crowded tournament selection, a pair of individuals is selected randomly, and the one with the lower rank is selected. If the ranks of the individuals are equal, then the individual with a larger crowding distance is chosen. The large crowding distance ensures that the solutions are spread along the Pareto-optimal front.

3.3 Genetic Operators

In MOGAMOD, the usual one-point crossover operator is stochastically applied with a predefined probability, using two individuals of the selected pool. The mutation operator is used to foster more exploration of the search space and to avoid unrecoverable loss of genetic material that leads to premature convergence to some local minima. In general, mutation is implemented by changing the value of a specific position of an individual with a given probability, denominated mutation probability. MOGAMOD developed three mutation operators tailored for our genome representation:

Shift the starting location towards the right: The value in the starting location of a randomly selected gene is increased by one.
Shift the starting location towards the left: The value in the starting location of a randomly selected gene is decreased by one.
Random-changing: The mutation produces a small integer number that is then added to or subtracted from the current content of any of length, weight or starting location. This is implemented in such a way that the lower and upper bounds the domain of the field are never exceeded.

To sum up, MOGAMOD process employed in this study can be summarized by the following algorithm,

The Algorithm:

Input: Population size N; Maximum number of generations G; Crossover probability p_c ; Mutation rate p_m.

Output: Nondominated set

(1) P:=Initialize (P)

(2) *while* the termination criterion is not satisfied *do*

(3) C:=Select From (P)

(4) C^l:=Genetic Operators (C)

(5) P:=Replace (PUC^l)

(6) *end while*

(7) return (P)

First, an initial population P is generated in Step 1. Pairs of parent solutions are chosen from the current population P in Step 3. The set of the selected pairs of parent solutions is denoted by C in Step 3. Crossover and mutation operations are applied to each pair in C to generate the offspring population C^l in Step 4. The next population is constructed by choosing good solutions from the merged population PUC^l. The pareto-dominance relation and a crowding measure are used to evaluate each solution in the current population P in Step 3 and the merged population PUC^l in Step 5. Elitism is implemented in Step 5 by choosing good solutions as members in the next population from the merged solution PUC^l.

4 Experimental Results

We conducted some experiments in order to analyze and demonstrate the efficiency and effectiveness of MOGAMOD. Further, the superiority of the proposed approach has been demonstrated by comparing it with three existing motif discovery methods, namely AlignACE [17], MEME [1], and Weeder [16]. In our experiments, we concentrate on testing the time requirements as well as changes in the main factors that affects the proposed multi-objective process, namely finding nondominated sets, support, length and similarity. All of the experiments have been conducted on a Pentium IV 3.0GHz CPU with 1GB of memory and running Windows XP. As data sets are concerned, we used two different data sets of sequences utilized as a benchmark for assessing computational tools for the discovery of transcription factor binding sites [24], which were selected from TRANSFAC database [25].

We concentrated our analysis on yst04r and yst08r sequence data sets. Further, in all the experiments conducted in this study, MOGAMOD process started with a population of 200 individuals. As the termination criteria, the maximum number of generations has been fixed at 3000. Finally, while the crossover probability is chosen to be 0.8, the mutation rate of 0.3 was used for each kind of mutation.

Table 3. The objective values of the nondominated solutions for yst04r

	Support	Length	Similarity
MOGAMOD	4	24	0.76
		20	0.78
		15	0.87
	5	15	0.82
		14^{*}	0.84
	6	14^{+}	0.77
		13	0.81
	7	9	0.80
		8	0.84
Single-objective GA	5	9	0.88

Table 4. Comparisons of the conserved motifs predicted by five different methods for yst04r

Method	**Predicted Motif**
AlignACE	CGGGATTCCA
MEME	CGGGATTCCCC
Weeder	TTTTCTGGCA
Single-objective GA	CTGGCATCC
MOGAMOD	*CGAGCTTCCACTAA $^{+}$CGGGATTCCTCTAT

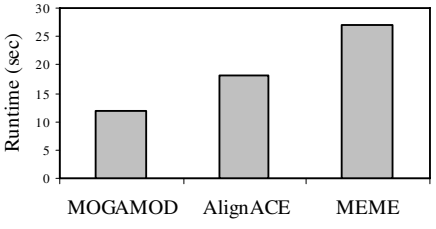


Fig. 2. Comparison of runtimes for yst04r data set

The standard single-objective GA was also executed using the same parameter values; and the weight values w_1=5, w_2=1 and w_3=1. Three sets of experiments for each data set were carried out. The first set of experiments is dedicated to evaluate the yst04r sequence data set. The data set contains 7 sequences of 1000 bps each. Some nondominated solutions found by MOGAMOD are reported in Table 3. Here, the values of length and similarity of some nondominated solutions are given for four different values of support. As can be easily seen from Table 3, as the support value increases, the motif length decreases. However, for each number of the supports, as the motif length decreases, the similarity value raises. Thus, the tradeoff between the

similarity and the motif length is clearly observed for four values of support. Table 3 gives the solution found by the standard single-objective GA process as well.

In the second experiment of the first set, we showed the consensus motif patterns obtained for five different motif discovery methods. Table 4 gives the results of this experiment. An important point here is that while MOGAMOD finds alternative solutions for different values of support, the other methods extract only one motif pattern. For example, CGAGCTTCCACTAA and CGGGATTCCTCTAT are two motifs predicted by MOGAMOD which have the same length but different support and similarity values. The final experiment for this set compares the runtimes of four different methods. The results are reported in Figure 2. The runtime reported for MOGAMOD represents a nondominated solution at the end of 3000 generations. As a result of this experiment, it has been observed that MOGAMOD outperforms the other two approaches for yst04r data set. The runtime of Weeder is not available.

Table 5. The objective values of the nondominated solutions for yst08r.

	Support	Length	Similarity
MOGAMOD	7	20	0.75
		15^{*1}	0.84
		15^{+1}	0.87
	8	15	0.79
		14^{*2}	0.83
		13^{+2}	0.85
	9	13	0.82
		12	0.84
	10	12	0.79
		11	0.82
	11	11	0.77
		11	0.80
Single-objective GA	8	10	0.86

In the second set of the experiments, we applied MOGAMOD on a data set having larger number of sequences, yst08r, as its sequence length is the same with the previous data set. This new data set contains 11 sequences. The first experiment obtains the values of objectives of the nondominated set for yst08r data set. Table 5 reports the results. As can be easily seen from Table 5, for two solutions having the same motif length, the similarity value of the solution whose support value is higher, is lower than that of the other. The second experiment deals with comparing the conserved motifs predicted by five different approaches. It can be easily realized from Table 6 that MOGAMOD and AlignACE produce multiple motifs.

Furthermore, Table 6 shows two multiple motifs discovered by MOGAMOD whose lengths are different, as well. The length of the first set of multiple motifs is 15 and their support value is 7 while the lengths of the second set of multiple motifs are 14 and 13, and their support value is 8.

Table 6. Comparisons of the conserved motifs predicted for yst08r

Method	Predicted Motif
AlignACE	CACCCAGACAC TGATTGCACTGA
MEME	CACCCAGACAC
Weeder	ACACCCAGAC
Single-objective GA	AACCCAGACA
MOGAMOD	^{*1}TCTGGCATCCAGTTT $^{+1}$GCGACTGGGTGCCTG ^{*2}GCCAGAAAAAGGCG $^{+2}$ACACCCAGACATC

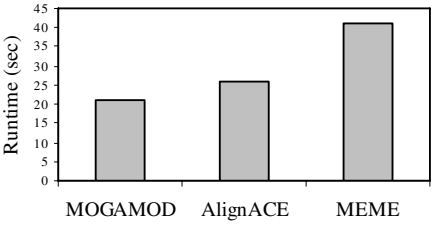


Fig. 3. Comparison of runtimes for yst08r data set

The third experiment investigates the runtimes of the proposed method and the other approaches in the case of a data set having large number of sequences. The results in Figure 3 demonstrate that the performance of MOGAMOD decreased with respect to the previous data set. This is an expected result since the length of individuals in population raises with increasing sequence number in the data set.

5 Discussion and Conclusions

In this paper, we contributed to the ongoing research by proposing a multi-objective GA based method for discovering optimized motifs (MOGAMOD) with respect to criteria we defined. These criteria are similarity of instances, predicted motif length and support exhibiting the strongness of motif.

MOGAMOD includes five contributions. First, the algorithm is equally applicable to any variety of sequential data. Second, it allows arbitrary similarity measure. Although we used relatively a simple similarity measure in the paper, it can be easily changed or extended. Another contribution is that a large number of nondominated sets are obtained by its single run. Thus, the decision maker can understand the tradeoff between the similarity, motif length and support by the obtained motifs. Next, by MOGAMOD, more than one instance may be discovered in the same sequence and multiple-motifs may be extracted. Fourth, the optimal motifs are obtained without

giving motif length. Finally, MOGAMOD outperforms the two well-known motif discovery methods in terms of runtime.

The experiments conducted on two data sets illustrated that the proposed approach produces meaningful results and has reasonable efficiency. The results of two data sets are consistent and hence encouraging. In the future, we will revise our similarity measure to make this score more realistic, improve our algorithm such that one could have better performance in lower similar sequences, and experiment with the realistic data sets having longer sequences and large number of sequences. Currently, we are also investigating how to find gapped motifs.

Acknowledgements

This study was supported by TUBITAK (The Scientific and Technological Research Council of Turkey) under Grant No: 106E199.

References

1. Bailey, T.L, Elkan, C.: Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In: Proc. Second Int. Conf. ISMB, USA, pp. 28–36 (1994)
2. Zhang, Y., Zaki, M.: EXMOTIF: Efficient structured motif extraction. Algorithms for Molecular Biology 1, 21 (2006)
3. Zhang, Y., Zaki, M.: SMOTIF: Efficient structured pattern and motif search. Algorithms for Molecular Biology 1, 22 (2006)
4. Pisanti, N., Carvalho, A.M., Marsan, L., Sagot, M.F.: RISOTTO: Fast extraction of motifs with mismatches. In: 7th Latin American Theoretical Informatics Symposium (2006)
5. Che, D., et al.: MDGA: motif discovery using a genetic algorithm. In: Proc. GECCO'05, USA, pp. 447–452 (2005)
6. Congdon, C.B., et al.: Preliminary results for GAMI: a genetic algorithms approach to motif inference. In: Proc. CIBCB'05, USA, pp. 1–8 (2005)
7. Deb, K., et al.: A fast and elitist multi-objective genetic algorithm: NSGA II. IEEE Trans. Evolutionary Computation 6, 182–197 (2002)
8. D'heaseleer, P.: What are DNA sequence motifs? Nat. Biotechnol 24, 423–425 (2006)
9. Kaya, M., Alhajj, R.: Integrating Multi-Objective Genetic Algorithms into Clustering for Fuzzy Association Rules Mining. In: IEEE International Conference on Data Mining (ICDM 2004), 1-4 November 2004, Brighton, UK (2004)
10. Kaya, M., Alhajj, R.: Multi-Objective Genetic Algorithm Based Approach for Optimizing Fuzzy Sequential Patterns. In: 16th IEEE International Conference on Tools with Artificial Intelligence, 15-17 November 2004, Boca Raton, FL, USA (2004)
11. Kaya, M.: Multi-Objective Genetic Algorithm Based Approaches for Mining Optimized Fuzzy Association Rules. Soft Computing Journal 10(7), 578–586 (2006)
12. Li, M., Ma, B., Wang, L.: Finding similar regions in many strings. In: Proc. STOC, USA, pp. 473–482 (1999)
13. Liu, F.M.M., et al.: FMGA: finding motifs by genetic algorithm. In: Proc. BIBE'04 Taiwan, pp. 459–466 (2004)
14. Notredame, C., Higgins, D.G.: SAGA: Sequence Alignment by Genetic Algorithm. Nucleic Acids Res. 24, 1515–1524 (1996)

15. Paul, T.K., Iba, H.: Identification of weak motifs in multiple biological sequences using genetic algorithm. In: Proc.GECCO'06, USA, pp. 271–278 (2006)
16. Pavesi, G., et al.: Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. Nucleic Acids Res. 32, W199–W203 (2004)
17. Roth, F.P., et al.: Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. Nat. Biotechnol. 16, 939–945 (1998)
18. Sinha, S., Tompa, M.: YMF: a program for discovery of novel transcription factor binding sites by statistical overrepresentation. Nucleic Acids Res. 31, 3586–3588 (2003)
19. Stine, M., et al.: Motif discovery in upstream sequences of coordinately expressed genes. In: CEC'03, USA, pp. 1596–1603 (2003)
20. Tatusova, T.A., Madden, T.L.: Blast2 sequences, a new tool for comparing protein and nucleotide sequences. FEMS Microbiology Letters 2, 247–250 (1999)
21. Thijs, G., et al.: A Gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. J. Comp. Biol. 9, 447–464 (2002)
22. Thompson, W., et al.: Gibbs Recursive Sampler: Finding transcription factor binding sites. J. Nucleic Acids Research 31, 3580–3585 (2003)
23. Tompa, M.: An exact method for finding short motifs in sequences with application to the ribosome binding site problem. In: Proc. Int. Conf. ISMB, Germany, pp. 262–271 (1999)
24. Tompa, M., et al.: Assessing computational tools for the discovery of transcription factor binding sites. Nat. Biotechnol. 23, 137–144 (2005)
25. Wingender, E., et al.: TRANSFAC: a database on transcription factors and their DNA binding sites. Nucleic Acids Research 24, 238–241 (1996)
26. Paul, T.K., Iba, H.: Identification of weak motifs in multiple biological sequences using genetic algorithm. In: Proc.GECCO'06, USA, pp. 271–278 (2006)
27. Zitzler, E., et al.: Comparison of multiobjective evolutionary algorithms: empirical results. Evolutionary Computation 2, 173–195 (2000)

Soft Topographic Map for Clustering and Classification of Bacteria

Massimo La Rosa[1], Giuseppe Di Fatta[2], Salvatore Gaglio[13], Giovanni M. Giammanco[4], Riccardo Rizzo[1], and Alfonso M. Urso[1]

[1] ICAR-CNR, Consiglio Nazionale delle Ricerche, Palermo, Italy
[2] School of Systems Engineering, University of Reading, UK
[3] Dipartimento di Ingegneria Informatica, Universitá di Palermo, Italy
[4] Dipartimento di Igiene e Microbiologia, Universitá di Palermo, Italy

Abstract. In this work a new method for clustering and building a topographic representation of a bacteria taxonomy is presented. The method is based on the analysis of stable parts of the genome, the so-called "housekeeping genes". The proposed method generates topographic maps of the bacteria taxonomy, where relations among different type strains can be visually inspected and verified. Two well known DNA alignement algorithms are applied to the genomic sequences. Topographic maps are optimized to represent the similarity among the sequences according to their evolutionary distances. The experimental analysis is carried out on 147 type strains of the Gammaprotebacteria class by means of the 16S rRNA housekeeping gene. Complete sequences of the gene have been retrieved from the NCBI public database. In the experimental tests the maps show clusters of homologous type strains and presents some singular cases potentially due to incorrect classification or erroneous annotations in the database.

1 Introduction

Microbial identification is crucial for the study of infectious diseases. The classical method to identify bacterial isolates is based on the comparison of morphologic and phenotypic characteristics to those described as type or typical strains. Recently a new naming approach based on bacteria genotype has been proposed and is currently under development. In this new approach phylogenetic relationships of bacteria could be determined by comparing a stable part of the genetic code. The part of the genetic code commonly used for taxonomic purposes for bacteria is the 16S rRNA "housekeeping" gene. The 16S rRNA gene sequence analysis can be used to obtain a classification for rare or poorly described bacteria, to classify organisms with an unusual phenotype in a well defined taxon, to find misclassification that can lead to the discovery and description of new pathogens.

The aim of this work is to obtain a topographic representation of bacteria clusters to visualize the relations among them. Moreover, we intend to achieve this objective by using directly the genotype information, without building a

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 332–343, 2007.

feature space. Many clustering approaches are based on a feature space where objects are represented. Biological datasets usually contain large objects (long nucleotides sequences or images); a vector space representation of such objects can be difficult and typically results in a high dimensional space where the euclidean distance is a low contrast metric. The definition of a vector space also requires the choice of a set of meaningful axes that represent some measurable qualities of the objects. In DNA sequences this approach is not straightforward and may be hindered by an arbitrary choice of features. According to these considerations we do not adopt a vector space representation, but a matrix of pairwise distances obtained directly from the genetic sequences. Such a matrix can be computed in terms of string distances by means of well understood and theoretically sound techniques commonly used in genomics.

The paper is organized as follows: in section 2 we refer to works that focus on similar classification problems of biological species; in sections 3 and 4 we describe the algorithms we have adopted for the similarity measure and the generation of topographic maps; in section 5 we present an experimental analysis of the proposed method and provide an interpretation of the results.

2 Related Work

In recent years, several attempts to reorganize actual bacteria taxonomy have been carried out by adopting 16S rRNA gene sequences. Authors in [1] focused on the study of bacteria belonging to the prokaryothic phyla and adopted the Principal Component Analysis method [2] on matrices of evolutionary distances. Clarridge [3], Drancourt et al. [4,5] carried out an analysis of 16S rRNA gene sequences to classify bacteria with atypical phenotype: they proposed that two bacterial isolates would belong to different species if the dissimilarity in the 16S rRNA gene sequences between them was more than 1% and less than 3%. Clustering approaches for DNA sequences [7] and for protein sequences [9] adopted Median Som, an extension of the Self-Organizing Map (SOM) to non-vectorial data. Chen et al. [11] proposed a protein sequence clustering method based on the Optic algorithm [12]. Butte and Kohane [8] described a technique to find functional genomic clusters in RNA expression data by computing the entropy of gene expression patterns and the mutual information between RNA expression patterns for each pair of genes. INPARANOID [13] is another related approach that performs a clustering based on BLAST [14] scores to find orthologs and in-paralogs in two species.

Among other algorithms for the clustering of pairwise proximity data, it is worth to mention an approach to segment textured images [29]. Dubnov et al. [16] proposed a nonparametric pairwise clustering algorithm that iteratively extracts the two most prominent clusters in the dataset, thus generating a hierarchical clustering structure. A hierarchical approach was also followed in [17,18]. Other works, e.g. [19,20], adopted Multidimensional Scaling [22] to embed dissimilarity data in a Euclidean space.

3 Genetic Sequence Similarity

3.1 Sequence Alignment

Sequence alignment allows to compare homologous sites of the same gene between two different species. For this purpose, we used two of the most popular alignment algorithms: ClustalW [23] for multiple-alignment; and Needleman-Wunsch [24] for pairwise alignment. The ClustalW algorithm aims to produce the best alignment configuration considering all the sequences at the same time, whereas Needleman-Wunsch algorithm provides a global optimum alignment between two sequences even of different length. Sequence alignment algorithms usually insert gaps in the input sequences in order to stretch them and to find the best matching configuration: gaps represent nucleotide insertions or deletions and are very important in terms of molecular evolution. An example of pairwise alignment is shown in Figure 1.

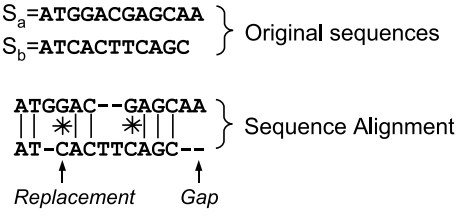


Fig. 1. Pairwise alignment between two gene sequences

3.2 Evolutionary Distance

The evolutionary distance is a distance measure between two homologous sequences, previously aligned. There are several kinds of evolutionary distances: the simplest one is the number of nucleotide substitutions per site. The number of substitutions observed between sequences is often smaller than the number of substitutions that have actually taken place. This is due to many genetic phenomena such as multiple substitutions on the same site (*multiple hits*), convergent substitutions or retro-mutations. As a consequence, it is important to use stochastic methods in order to obtain an exact estimate of evolutionary distances. Many stochastic models exist that differ from each other on the basis of their a priori assumptions.

The most common a priori assumptions are:

- all sites evolve in an independent manner;
- all sites can change with the same probability;
- all kinds of substitution are equally probable;
- substitution speed is constant over time.

In our study, we used the method proposed by Jukes and Cantor [25], where all the assumptions above are valid. According to [25], the evolutionary distance d between two nucleotide sequences is equal to:

$$d = -\frac{3}{4}\ln\left(1 - \frac{4}{3}p\right), \tag{1}$$

where p is the number of substitutions per site, defined as:

$$p = \frac{\text{number of different nucleotides}}{\text{total number of compared nucleotides}}. \tag{2}$$

It is important to note that sites containing gaps or undefined bases are not considered in the computation of distances.

Evolutionary distances computed with (1) constitute the elements of a dissimilarity matrix that represents the input for the algorithm described in the next section.

4 Soft Topographic Map Algorithm

A widely used algorithm for topographic maps is the Kohonen's Self Organizing Map (SOM) algorithm [31], but it does not operate with dissimilarity data.

According to Luttrell's work [26], the generation of topographic maps can be interpreted as an optimization problem based on the minimization of a cost function. This cost function represents an energy function and takes its minimum when each data point is mapped to the best matching neuron, thus providing the optimal set of parameters for the map.

An algorithm based on this formulation of the problem was developed by Graepel, Burger and Obermayer [27,28] and provides an extension of SOM to arbitrary distance measures. This algorithm is called Soft Topographic Map (STM) and creates a map using a set of units (neurons or models) organized in a rectangular lattice that defines their neighbourhood relationships.

The cost function for soft topographic mapping of proximity data (in our case a dissimilarity matrix) can be formulated as follows:

$$E(\{c_{t\mathbf{r}}\}) = \frac{1}{2}\sum_{t,t'}\sum_{\mathbf{r},\mathbf{s},\mathbf{u}} \frac{c_{t\mathbf{r}}h_{\mathbf{rs}}c_{t'\mathbf{u}}h_{\mathbf{us}}}{\sum_{t''}\sum_{\mathbf{v}} c_{t''\mathbf{v}}h_{\mathbf{vs}}} d_{tt'}, \tag{3}$$

where $d_{tt'}$ is the generic element of the dissimilarity matrix, namely the pairwise distance among nucleotide sequences of bacteria t and t'. Two constraints hold in (3): $\sum_{\mathbf{r}} c_{t\mathbf{r}} = 1, \forall t$, i.e. each data vector can belong only to one neuron $\mathbf{r}$, and $\sum_{\mathbf{s}} h_{\mathbf{rs}} = 1, \forall \mathbf{r}$. The function $h_{\mathbf{rs}}$ is equivalent to the neighborhood function of classic SOM algorithm and represents the coupling between neurons $\mathbf{r}$ and $\mathbf{s}$ in the map grid. $h_{\mathbf{rs}}$ is usually chosen as a normalized Gaussian function such as:

$$h_{\mathbf{rs}} \propto \exp\left(-\frac{|\mathbf{r}-\mathbf{s}|^2}{2\sigma^2}\right), \forall \mathbf{r}, \mathbf{s}. \tag{4}$$

Table 1. Soft Topographic Map algorithm

1. Initialization Step:
   (a) $e_{t\mathbf{r}} \leftarrow n_{t\mathbf{r}}, \forall t, \mathbf{r}, n_{t\mathbf{r}} \in [0, 1]$
   (b) compute lookup table for $h_{\mathbf{rs}}$ as in Eq. (4)
   (c) compute dissimilarity matrix from input data as in Eq. (1)
   (d) put $\beta \cong \beta^*$
   (e) choose β_{final}, increasing temperature factor η, convergence threshold ϵ
2. Training Step:
   (a) while $\beta < \beta_{final}$ (Annealing cycle)
      i. repeat (EM cycle)
         A. E step: compute $P(\mathbf{x}_t \in \mathbf{C_r})\forall t, \mathbf{r}$ as in Eq. (5)
         B. M step: compute $a_{t\mathbf{r}}^{new}, \forall t, \mathbf{r}$ as in Eq. (7)
         C. M step: compute $e_{t\mathbf{r}}^{new}, \forall t, \mathbf{r}$ as in Eq. (6)
      ii. until $\left\| e_{t\mathbf{r}}^{new} - e_{t\mathbf{r}}^{old} \right\| < \epsilon$
      iii. put $\beta \leftarrow \eta\beta$
   (b) end while

In order to optimize the cost function the deterministic annealing [29,30] technique has been used. This technique is based on the optimization of a family of cost functions, representing free energy, that depend on the parameter β, the so called inverse temperature. This parameter represents the amount of smoothing that is done to the original cost function.

The minimization of this function leads to the probability of the assignment of the data vector t to the node $\mathbf{r}$ (i.e. to its cluster $\mathbf{C_r}$):

$$P(\mathbf{x}_t \in \mathbf{C_r}) = \frac{\exp(-\beta e_{t\mathbf{r}})}{\sum_{\mathbf{u}} \exp(-\beta e_{t\mathbf{u}})}, \forall t, \mathbf{r}. \tag{5}$$

In Equation (5), $e_{t\mathbf{r}}$ is the partial assignment cost of data vector $\mathbf{x}_t$ to be assigned to cluster $\mathbf{C_r}$, and it is defined as:

$$e_{t\mathbf{r}} = \sum_{\mathbf{s}} h_{\mathbf{rs}} \sum_{t'} a_{t'\mathbf{s}} \left(d_{tt'} - \frac{1}{2} \sum_{t''} a_{t''\mathbf{s}} d_{t't''} \right), \forall t, \mathbf{r}. \tag{6}$$

Equation (6) is obtained considering that diagonal elements of the dissimilarity matrix are equal to zero and that the dissimilarity matrix is symmetric. The weighting factors $a_{t\mathbf{r}}$ are given by:

$$a_{t\mathbf{r}} = \frac{\sum_{\mathbf{s}} h_{\mathbf{rs}} P(\mathbf{x}_t \in \mathbf{C_s})}{\sum_{t'} \sum_{\mathbf{s}} h_{\mathbf{rs}} P(\mathbf{x}_{t'} \in \mathbf{C_s})}, \forall t, \mathbf{r} \tag{7}$$

and can be seen as weighted averages over data vectors.

The Soft Topographic Map algorithm for proximity data described above can be summarized in the pseudo code of Table 1. Minimization procedure can be done in two steps, formed by two nested loops. The inner loop 2(a)i constitutes an

expectation-maximization (EM) algorithm: starting from a random initialization of partial costs, equations (5), (7), (6) are computed in sequence for a fixed value of β until the difference between current partial costs and previous partial costs is lower than a certain threshold. Then, in the outer loop 2a, in order to find the global minimum of the cost function, β is gradually increased and the inner loop repeated. β is increased according to the annealing scheme $\beta \leftarrow \eta\beta$, with $\eta = 1.1 \ldots 2.0$, up to a previously chosen β_{final}.

As seen in [27], the initial value of β should be just above a certain value β^* calculated as:

$$\beta^* = \frac{1}{\lambda^{\mathbf{C}}_{max}\lambda^{\mathbf{G}}_{max}}, \tag{8}$$

where $\lambda^{\mathbf{C}}_{max}$ is the largest eigenvalue of the covariance matrix $\mathbf{C}$ of the data and $\lambda^{\mathbf{G}}_{max}$ is the largest eigenvalue of a matrix $\mathbf{G}$, whose elements are equal to:

$$g_{\mathbf{rt}} = \sum_{\mathbf{s}} h_{\mathbf{rs}} \left(h_{\mathbf{st}} - \frac{1}{M} \right). \tag{9}$$

5 Experimental Analysis

5.1 Bacteria Dataset

In order to test our approach, we have built a database of 16S rRNA bacteria gene sequences. The choice of the bacteria set has been done according to the current taxonomy [1]. We focused on the bacteria belonging to Phylum BXII, Proteobacteria; Class III, Gammaproteobacteria: this class includes some of the most common and dangerous bacteria related to human pathologies. In the Gammaproteobacteria class there are 14 orders, each of them containing one or more family. Each family is divided in genera; for each genus we selected the type strains, as shown in Figure 2.

For each type strain we selected the 16S rRNA gene sequence, which contains approximately 1400 nucleotides. The resulting 147 sequences were retrieved from GenBank [33] in FASTA format [15].

Each gene sequence is labelled according to its order in the actual taxonomy.

5.2 Experimental Results

We carried out a set of experimental tests using the algorithm described in the section 4 with the bacteria dataset of section 5.1. We used both the dissimilarity matrices obtained from multiple alignment of sequences and pairwise alignment of sequences in order to compare the results. More specifically, we used two well known bioinformatic tools: Mega software [34], that implements ClustalW algorithm, and Emboss tools [35] for Needleman-Wunsch algorithm. In both situations, we used default options.

We applied a slightly tuned version of Soft Topographic Map algorithm: in order to speed up processing time, neighbourhood functions associated to each neuron have been set to zero if they referred to neurons outside a previously

		Order Name	Number of Families	Number of Type Strains
Gammaproteobacteria	□	Chromatiales	3 Families	25 Type Strains
	△	Xanthomonodales	1 Family	11 Type Strains
	●	Thiotrichales	3 Families	11 Type Strains
	▣	Methylococcales	1 Families	7 Type Strains
	◎	Pseudomonadales	2 Families	7 Type Strains
	★	Vibrionales	1 Family	3 Type Strains
	◉	Enterobacteriales	1 Family	39 Type Strains
	○	Acidithobacillales	2 Families	2 Type Strains
	■	Cardiobacteriales	1 Family	3 Type Strains
	▲	Legionellales	2 Families	3 Type Strains
	◎	Oceanospirillales	4 Families	11 Type Strains
	⊠	Alteromonadales	1 Family	13 Type Strains
	☆	Aeromonadales	2 Families	7 Type Strains
	◬	Pasteurellales	1 Families	6 Type Strains
				147 Type Strains

Fig. 2. Actual taxonomy of the bacteria dataset

chosen radius in the grid. The radius has been put to 1/3 of the side of maps. As for the other parameters of the algorithm, we put the annealing increasing factor $\eta = 1.1$, and threshold convergence $\epsilon = 10^{-5}$, as suggested by [27]. After several tests we chose, as a good compromise between processing time and clustering quality, the final value of inverse temperature equal to 10 times the initial value, leading as a consequence 25 learning epochs; finally we put the width of neighbourhood functions σ to 0.5.

We generated several maps of different dimensions, from 8×8 up to 20×20 neurons. The dimensions of the maps were set by considering the number of input patterns (147 gene sequences) and the number of expected clusters (14 orders in the taxonomy). We compared each pair of maps of the same dimension obtained from multiple alignment and pairwise alignment. The results were quite similar and we can state that the alignment technique does not affect final results.

In Figures 3, 4, 5, we show the results provided by 12×12, 16×16, 20×20 maps, trained with the dissimilarity matrix using the pairwise alignment. In the maps, bright areas denote proximity and dark zones represent distance, according to the U-Matrix style [32].

It should be noticed that in larger maps the units tend to classify homogeneous patterns better. Namely, comparing the maps we can observe that the number of bacteria belonging to mixed clusters, i.e units containing bacteria of different orders, decreases as the number of neurons increases (Figure 6). Therefore, the 20×20 map is the most accurate. In all the maps most of the bacteria are classified according to their order in the actual taxonomy. We can also see that

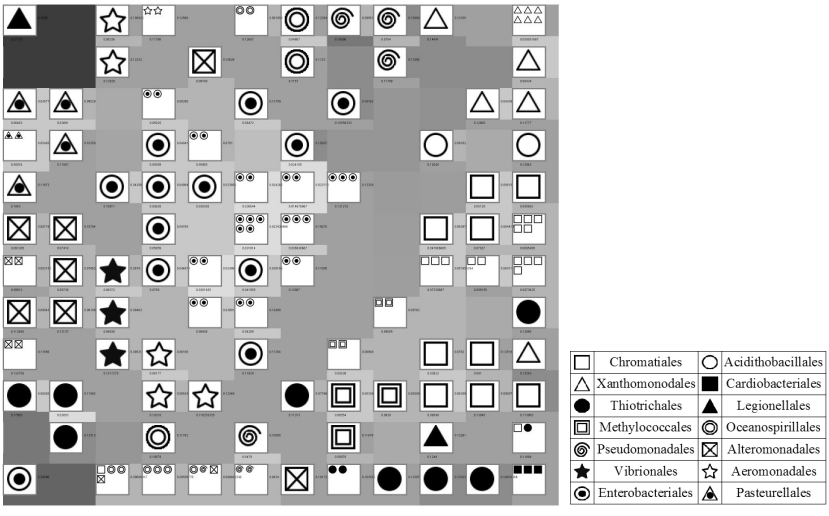


Fig. 3. 12 × 12 topographic map of bacteria dataset

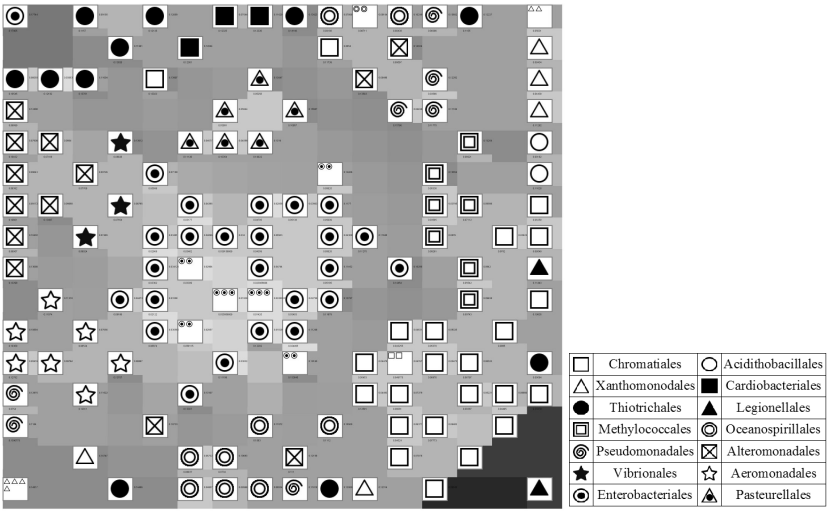


Fig. 4. 16 × 16 topographic map of bacteria dataset

bacteria belonging to the order "Enterobacteriales" are split into a series of adjacent clusters in the central part of the map. This could mean that the order "Enterobacteriales" could be subdivided into distinct families rather than the single one of the actual taxonomy (see Figure 2).

Finally, an interesting result is that there are some anomalies that are constant for all the tests regardless of the chosen map dimension and alignment algorithm.

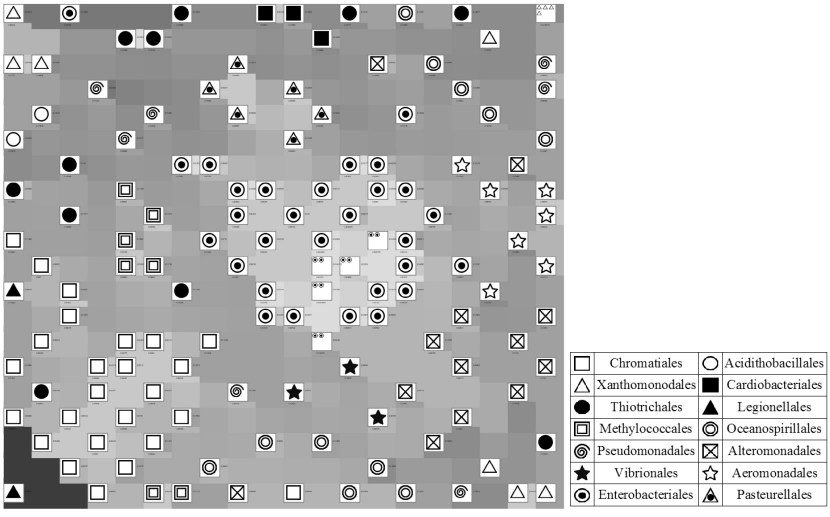


Fig. 5. 20 × 20 topographic map of bacteria dataset

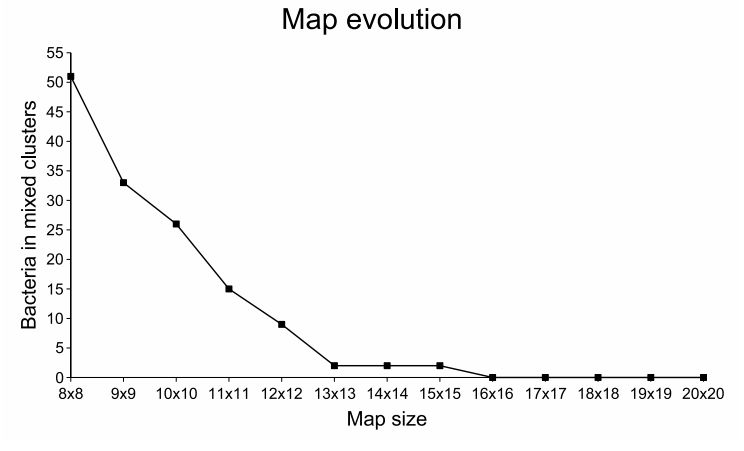


Fig. 6. Bacteria in mixed clusters w.r.t. map size

For example, in small maps (not shown here) the "Alterococcus agarolyticus" bacterium of the "Enterobacteriales" order is incorrectly clustered together with bacteria of other orders, whereas in larger maps it is isolated in an individual cluster, usually at the border of the map (e.g. at the lower left corner of Figure 3 and at upper left conrner of Figures 4 and 5). Another interesting example is given by "Legionella pneumophila" bacterium of "Legionellales" order: that in all maps is located in a corner of the grid and surrounded by a dark grey area. This would suggest that it can be considered to have an order of its own. In

general, we noticed that in the transition from smaller maps to larger ones there is always a set of bacteria that show the following anomalies:

- bacteria belonging to mixed clusters and far from their homologous bacteria,
- isolated bacteria in a single cluster far from their homologous bacteria.

In the former case, it is possible that those bacteria were either incorrectly classified or incorrectly registered into GenBank. In the latter, it is very likely that those bacteria could form new orders or families that have not been discovered by analyzing only phenotypic features.

In conclusion, although the topographic maps have shown a clustering that generally reflects the current taxonomy, some singular cases have been detected. The proposed approach is a first attempt to provide an innovative tool to support the correction of genetic sequence submission systems (e.g. GenBank) and to build a genotypic features based taxonomy.

6 Conclusions

In recent trends for the definition of bacteria taxonomy, genotypical characteristics are considered very important and type strains are compared on the basis of the stable part of the genetic code. In this paper the Soft Topographic Map algorithm has been applied to the clustering and classification of bacteria according to their genotypic similarity. In the similarity measure we have adopted the 16S rRNA gene sequence, as commonly used for taxonomic purposes. A characteristic of the proposed approach is that the topographic map is built directly from the genetic data, without using a vector space representation. The generated maps show that the proposed approach provides a clustering that generally reflects the current taxonomy with some singular cases. The map allows an easy identification of cases that could represent incorrect classification or incorrect registration in the database. In future research activities we intend to extend the analysis to other "housekeeping" genes and to combine different genotypical characteristics in order to obtain finer clustering and classification.

References

1. Garrity, G.M., Julia, B.A., Lilburn, T.: The revised road map to the manual. In: Garrity, G.M. (ed.) Bergey's manual of systematic bacteriology, pp. 159–187. Springer, New York (2004)
2. Joliffe, I.T.: Principal Component Analysis. Springer, New York (1986)
3. Clarridge III, J.E.: Impact of 16S rRNA Gene Sequence Analysis for Identification of Bacteria on Clinical Microbiology and Infectious Diseases. Clin. Microbiol. Rev. 17, 840–862 (2004)
4. Drancourt, M., Bollet, C., Carlioz, A., Martelin, R., Gayral, J.-P., Raoult, D.: 16S Ribosomal DNA Sequence Analysis of a Large Collection of Environmental and Clinical Unidentifiable Bacterial Isolates. J. Clin. Microbiol 38, 3623–3630 (2000)

5. Drancourt, M., Berger, P., Raoult, D.: Systematic 16S rRNA Gene Sequencing of Atypical Clinical Isolates Identified 27 New Bacterial Species Associated with Humans. J. Clin. Microbiol. 42, 2197–2202 (2004)
6. Drancourt, M., Raoult, D.: Sequence-Based Identification of New Bacteria: a Proposition for Creation of an Orphan Bacterium Repository. J. Clin. Microbiol. 43, 4311–4315 (2005)
7. Oja, M., Somervuo, P., Kaski, S., Kohonen, T.: Clustering of human endogenous retrovirus sequences with median self-organizing map. In: WSOM'03. Workshop on Self-Organizing Maps (9-14 September 2003)
8. Butte, A.J., Kohane, I.S.: Mutual information relevance networks: functional genomics clustering using pairwise entropy measurements. In: Proc. Pacific Symposium on Biocomputing, vol. 5, pp. 415–426 (2000)
9. Somervuo, P., Kohonen, T.: Clustering and visualization of large protein sequence databases by means of an extension of the self-organizing map. In: Discovery Science. Proceedings of the Third International Conference, pp. 76–85 (2000)
10. Kohonen, T., Somervuo, P.: How to make large self-organizing maps for nonvectorial data. Neural Networks 15(8-9), 945–952 (2002)
11. Chen, Y., Reilly, K.D., Sprague, A.P., Guan, Z.: SEQOPTICS: A Protein Sequence Clustering Method. In: First International Multi-Symposiums on Computer and Computational Sciences. IMSCCS '06, 20-24 June 2006, vol. 1, pp. 69–75 (2006)
12. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: Ordering Points To Identify the Clustering Structure. In: SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, Philadelphia, Pennsylvania, USA, June 1-3, 1999, pp. 49–60 (1999)
13. Remm, M., Storm, C.E.V., Sonnhammer, E.L.L.: Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. Journal of Molecular Biology 314(5), 1041–1052 (2001)
14. Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic local alignment search tool. J. Mol. Biol. 232, 584–599 (1993)
15. http://www.ncbi.nlm.nih.gov/blast/fasta.shtml
16. Dubnov, S., El-Yaniv, R., Gdalyahu, Y., Schneidman, E., Tishby, N., Yona, G.: A new nonparametric pairwise clustering algorithm based on iterative estimation of distance profiles. Machine Learning 47, 35–61 (2002)
17. Buhmann, J., Zoller, T.: Active Learning for Hierarchical Pairwise Data Clustering. icpr, 2186 (2000)
18. Hofmann, T., Buhmann, J.M.: Hierarchical pairwise data clustering by mean–field annealing. In: Proceedings of ICANN'95, NEURON IMES'95, vol. II, pp. 197–202. EC2 & Cie (1995)
19. Graepel, T., Herbrich, R., Bollmann-Sdorra, P., Obermayer, K.: Classification on Pairwise Proximity Data. In: NIPS
20. Hofmann, T., Buhmann, J.: Multidimensional scaling and data clustering. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) Advances in Neural Information Processing Systems, vol. 7, pp. 459–466. MIT Press, Cambridge, Mass (1995)
21. Klock, H., Buhmann, J.M.: Multidimensional scaling by deterministic annealing. In: Pelillo, M., Hancock, E.R. (eds.) EMMCVPR 1997. LNCS, vol. 1223, pp. 246–260. Springer, Heidelberg (1997)
22. Torgerson, W.S.: Multidimensional scaling: I. Theory and method. Psychometrika 17, 401–419 (1952)

23. Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Research 22, 4673–4680 (1994)
24. Needleman, S.B., Wunsch, C.D.: J. Mol. Biol. 48, 443–453 (1970)
25. Jukes, T.H., Cantor, C.R.: Mammalian Protein Metabolism. In: Munro, H.N. (ed.) Evolution of Protein Molecules, pp. 21–132. Academic Press, New York (1969)
26. Luttrell, S.P.: A Bayesian analysis of self-organizing maps. Neural Comput. 6, 767–794 (1994)
27. Graepel, T., Burger, M., Obermayer, K.: Self-organizing maps: generalizations and new optimization techniques. Neurocomputing 21, 173–190 (1998)
28. Graepel, T., Obermayer, K.: A stochastic self organizing map for proximity data. Neural Computation 11, 139–155 (1999)
29. Hofmann, T., Buhmann, J.M.: Pairwise data clustering by deterministic annealing. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 1–14 (1997)
30. Rose, K.: Deterministic Annealing for Clustering, Compression, Classification, Regression, and Related Optimization Problems. Proc. of the IEEE 86(11), 2210–2239 (1998)
31. Kohonen, T.: Self-organizing maps. Springer, Heidelberg (1995)
32. Ultsch, A.: U*-Matrix: a Tool to visualize Clusters in high dimensional Data, Technical Report No. 36, Dept. of Mathematics and Computer Science, University of Marburg, Germany (2003)
33. http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Nucleotide
34. Kumar, S., Tamura, K., Nei, M.: MEGA3: Integrated software for Molecular Evolutionary Genetics Analysis and sequence alignment. Briefings in Bioinformatics 5, 150–163 (2004)
35. Rice, P., Longden, I., Bleasby, A.: EMBOSS: The European Molecular Biology Open Software Suite. Trends in Genetics 16(6), 276–277 (2000)

Fuzzy Logic Based Gait Classification for Hemiplegic Patients

A. Yardimci

Department of Industrial Automation, TBMYO, Akdeniz University,
Antalya, Turkey
yardimci@akdeniz.edu.tr

Abstract. In this study a fuzzy logic classification system was used first to discriminate healthy subjects from patients rather than classifying those using Brunnstrom stages. Decision making was performed in two stages: feature extraction of gait signals and the fuzzy logic classification system which is used Tsukamato-type inference method. According to our signal feature extraction studies, we focused on temporal events and symetrical features of gait signal. Developed system has six inputs while four of them for temporal features evaluation rule block and two of them symmetrical features evaluation rule block. Our simulation test results showed that proposed system classify correctly 100% of subjects as patient and healthy elderly. The correlation coefficient was found 0.85 for classification to subjects to correct Brunnstrom stages. The results show that classifying patients becomes increasingly difficult linearly according to hemiplegia's severity.

Keywords. Fuzzy logic classification, rehabilitation, hemiplegic gait.

1 Introduction

Gait quantification remains essential for monitoring and functional recovery during the rehabilitation process. Motor system recovery is quantitatively classified into Brunnstrom method's six stages [1]. Synergistic movements convert to non-synergic movements between Brunstromm stages III and IV. Thus, isolated joint movements actually move in unison. Post-stroke hemiplegic patients with Brunnstrom stage III show maximum spasticity that decreases with motor system recovery. A common procedure starts when a physiotherapist empirically observes the patient's gait and evaluates the rehabilitation training's effectiveness. Previous studies carried more precise kinetic and kinematic analysis using optoelectronic systems and forceplates [2], [3]. These systems, although powerful and flexible, are expensive and require both high technical skills and specialized experience to operate. The system must offer repeatability and flexibility (i.e. portable data acquisition system) based on accelerometry. We've worked on gait quantification in healthy elderly subjects and post-stroke hemiplegic (PSH) patients using both accelerometry techniques and advanced signal processing methods. We will explain our latest work on hemiplegic patient gait's classification with fuzzy logic (FL) using accelerometer-based

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 344–354, 2007.

acceleration. FL was discovered in 1965 by Zadeh [4]. Zadeh's publication of a 1969 paper applying fuzzy sets to biology was the first application of fuzzy logic to the medical field. The study aims to assess gait after stroke for both diagnosis support and therapy applications. Decision-making was performed in the gait signal's feature extraction and FL system using Tsukamato-type inference method. According to our signal feature extraction studies, we focused on the gait signal's temporal events (TE) and symmetrical features (SF). The developed FL classification system contains six inputs: four for TE rule block and two for SF rule block. The system classifies subject gaits as either healthy or hemiplegic gait. Additionally, the system matches hemiplegic gait to correct Brunnstrom stages. The signal feature extraction determined the FL system inputs upon examining the four hemiplegic gait acceleration (HGA) signal types at each Brunnstrom stage and healthy elderly gait signal. The entire rule blocks and some membership functions were rearranged to improve diagnostic and classification accuracy. We obtained conclusions concerning feature saliency of HGA signal classification via FL system analysis. Classification accuracies evaluated our system's performance. Our results confirmed our proposed model's potential in classifying HGA signals. The accelerometry technique is widely used clinically to investigate gait's body motion in healthy and PSH patients. Gait analysis via accelerometric records is one of the most important tools for diagnosing locomotion defects. Daily activity classification (standing, sitting, lying, and walking) was previously done using combined DC and AC acceleration signals.

Quantitative data gait analysis was traditionally a challenging endeavor. The main challenges were high dimensionality, temporal dependence, high variability, and nonlinear relationships. Responding to these challenges, summary statistics (e.g. mean, variance, correlations) and wave-form parameterizations (e.g. peak, amplitude), provided limited additional insight into gait data beyond what was observable from bivariate plots. It is now recognized that there's a lack of effective and robust techniques to reduce gait data [5, 6] and to extract information from highly-correlated gait data variables [7]. Researchers sought out novel ways to manipulate and interpret gait data. These new ventures originated from ideas from computer science, psychology, cognitive science, physics, and engineering. Sekine et al. demonstrated that the wavelet transform method effectively classified walking types for young subjects (but not for elderly subjects since gait changes with age) [8]. Locomotion is reduced for the elderly, compared to young people. The impact acceleration's amplitude of a heal strike decreases with age. The elderly also shuffle during locomotion, increasing the acceleration's complexity. Thus, it is extremely difficult classifying the PSH patient's walking type by using previous methods.

Fuzzy analysis treats variability as non-probabilistic uncertainty while statistical methods view variability as probabilistic randomness. Fuzzy set theory plays an important role in dealing with uncertainty when making decisions in medical applications. Thus, fuzzy sets continue attracting growing attention and interest in modern information technology, production technique, decision-making, pattern recognition diagnostics, data analysis [9-11]. Successful implementations of FL in biomedical engineering were previously reported for classification [12-16].

2 Methods

2.1 Measurement System

Figure 1 shows a schematic diagram of the measurement system [8]. An accelerometer device was constructed from three uni-axial accelerometers (type 3031-010, IC-Sensors, USA, size: 4x4x3mm; weight: 0.3 g; range: ±10 g; frequency response: 0-500 Hz). These were mounted orthogonally to record signals in the anteroposterior (x), lateral (y), and vertical (z) directions. The accelerometers were calibrated by measuring their outputs under controlled inclination. After the accelerometer device had been calibrated it was fixed on an acrylic plate that had two slits for a waist belt. It was fastened by an elastic waist belt to the subject's back in the lumbosacral region of the vertebral column, close to the subject's center of gravity while standing [8]. The accelerometer unit was connected to a portable data logger (Micro 8, Shimadzu, Japan) via an interface circuit. This data logger consisted of a CPU, a 10-bit A/D converter, an IC card interface, and a removable 2-MB IC memory card as shown in Figure 1. The interface circuit included three amplifiers and three second-order analog Butterworth lowpass filters as an anti-aliasing filter for each direction. The cutoff frequency was 500 Hz. The accelerometer outputs were digitized at a sampling rate 1024 Hz by the data logger and were recorded on the IC memory card. After the measurements were completed, the data was transferred via a card reader to a personal computer for further analysis.

2.2 Experimental Design

The experiments were performed with 28 poststroke hemiplegic (PSH) patients (19 male, 9 female, age 66± 11 years, height 1.54 ± 0.09 m, weight 54.6 ± 9.4 kg; mean ± SD). In this study, the PSH patients who participated were Brunnstrom stage III to VI (III: 12, IV: 9, V: 4 and VI: 3). The patients walked along a corridor at free speed with a cane and/or short leg brace, which were used during physical training. For comparison, seven healthy elderly subjects (two male, eight female, age 61 ± 5.1 years, height 1.49 ± 0.05 m, weight 49.7 ± 1.6kg; mean ± SD) participated in this study. The healthy elderly subjects walked at free speed, wearing their own shoes and without any special instructions. This study was approved by the local ethics committee, and all the subjects gave written informed consent before examination.

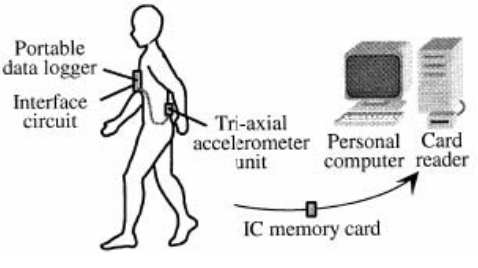


Fig. 1. Schematic diagram of measurement system

3 Classification of Hemiplegic Gate with Fuzzy Logic

Walking is a complicated cyclic set of movements requiring the coordination of many muscles. Repetitive motion is achieved via this coordination. An accurate and detailed mathematical model of this system is unavailable due to limited knowledge of the physiological system, making it impossible to apply traditional methods. We used statistical analysis to produce our own inputs plus membership functions before developing a fuzzy logic system capable of classifying hemiplegic gate. The gait signal's temporal features are walking speed, stride period, cadence, stride length, stance period, swing period, stance/swing ratio and double support. The gait signal's measurable features are walking speed, cadence, step length, double step length, step time difference, and double step time difference. Measurement and variable analysis are unable to characterize hemiplegic patients' erratic locomotion [17] because the relevant temporal information in hemiplegic gaits is included while measuring the walking speed. Prior studies revealed that the hemiplegic gate's temporal variables relate to motor recovery according to defined stages. Hemiplegic patients, even those with good motor recovery, by comparison walked more slowly. The patient's clinical

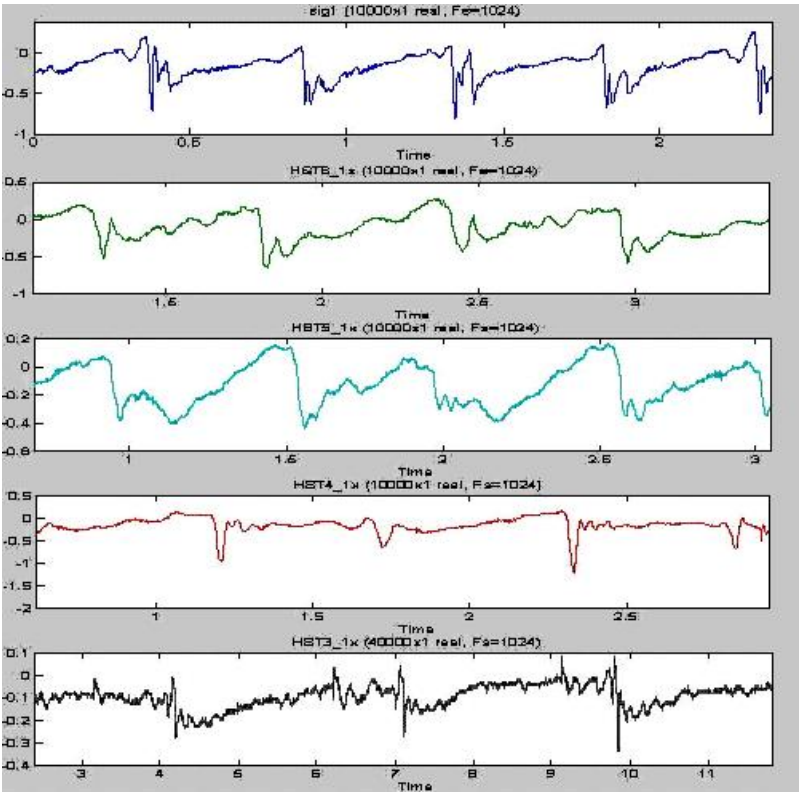


Fig. 2. Gait acceleraiton signals, top to bottom; Healthy STVI, STV, STIV, STIII

status matched the walking speed, which slowed down with the motor deficit's increasing severity. Investigations [17-19] revealed that walking speed is an important temporal variable of hemiplegic gait. Several algorithms exist to compute step times and quantify symmetry [20,21].

As previously mentioned, determining the classification system's inputs, we conducted several statistical analyses on gait features. Figures 2 and 3 show our results comparing walking speed: Figure 2 shows anteroposterior acceleration signals of elderly subjects, the top channel belonging to elderly patients while the bottom belongs to Brunstromm stage III patients.

Figure 3 shows step time comparison of healthy patients and various hemiplegic stage patients. We statistically examined all acceleration signals, anteroposterior (x), lateral (y), and vertical (z). Figures 4 and 5 illustrate the anteroposterior signal mean amplitudes of each subject and each group, respectively. Brevity prevents us from publishing all statistical results. We defined our fuzzy system structure after comparing our results. The fuzzy system structure identifies the fuzzy logic inference flow from input variables to output variables. The input interface's fuzzification translates analog inputs into fuzzy values. The fuzzy inference starts in rule blocks that contain linguistic control rules. The rule block's outputs are linguistic variables. The defuzzification in the output interfaces translates them into analog variables.

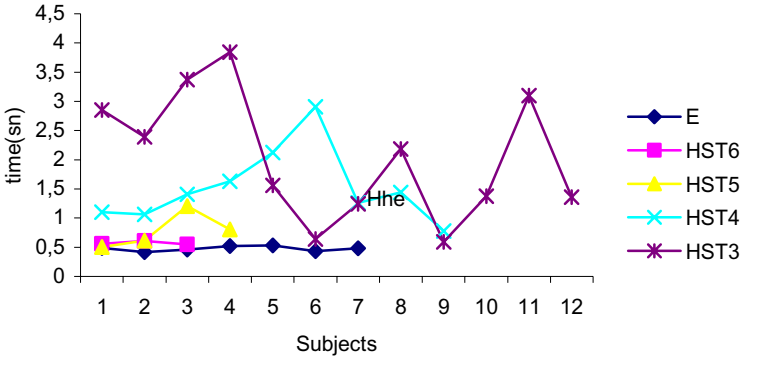


Fig. 3. Step time comparison of subjects (E:Elderly, HST6: Brunnstrom stage VI, HST5: Brunnstrom stageV, HST4:Brunnstrom stage IV, HST3:Brunnstrom stage III)

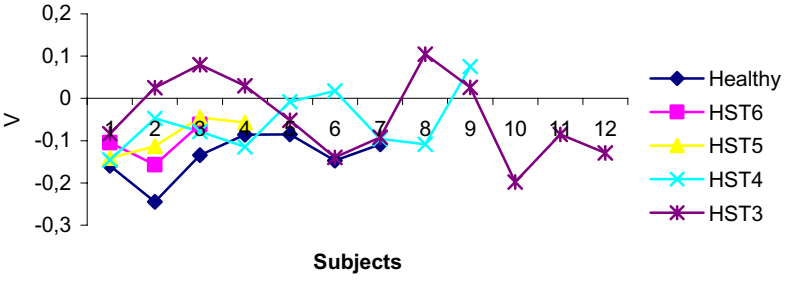


Fig. 4. Anteroposterior signal mean amplitude values for each subject

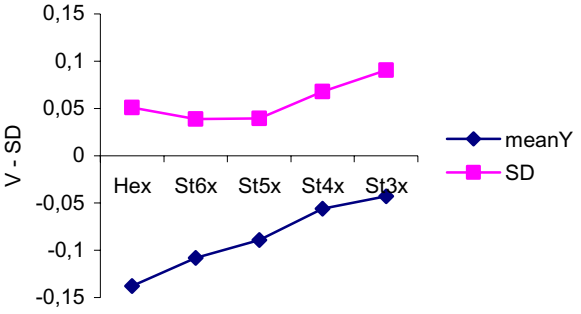


Fig. 5. Anteroposterior signal mean amplitudes of each groups (SD:Standart deviation)

Figure 6 shows the entire fuzzy system including input and output interfaces, and rule blocks. The connecting lines signify data flow.

The developed system contains two blocks: one evaluates signal amplitude features and contains four inputs; XAR, XAM, YVR, and ZLR. The other block evaluates signal symmetries and contains two inputs: XASI and XAST. These symbols represent the following variables:

XAR :Anteroposterior amplitude
XAM :Anteroposterior mean amplitude
YVR :Vertical amplitude
ZLR :Lateral amplitude
XASI :Anteroposterior signal symmetry index
XAST :Anteroposterior step time

All input membership functions contain three linguistic terms. The rule bases contain the fuzzy logic system's control strategy. The rules 'if' part describes the situation. The 'then' part describes the fuzzy system's response to the situation. The rules use the degree of support method to weigh each rule according to its importance. The first rule block contains 81 rules while the second contains only 9.

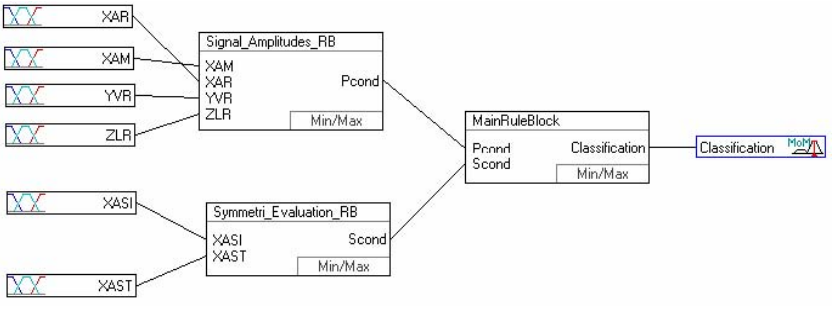


Fig. 6. Structure of the Fuzzy Logic classification

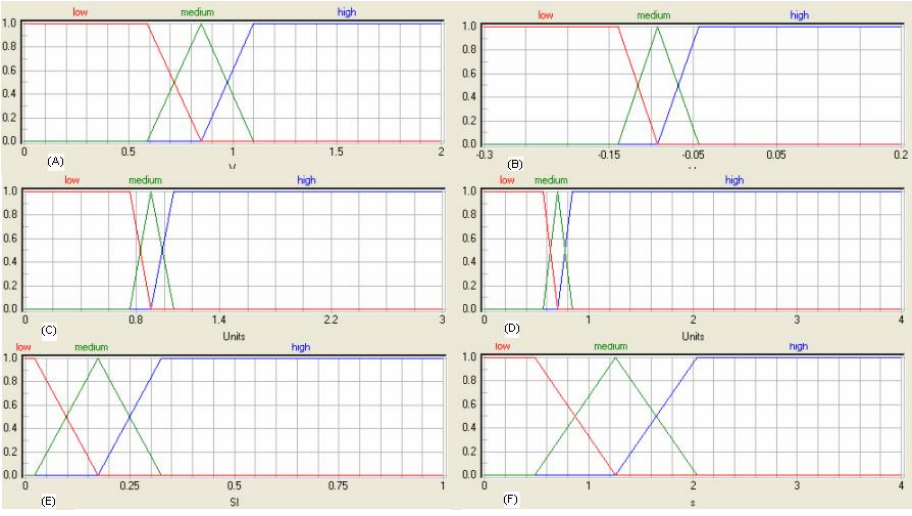


Fig. 7. Input membership functions (A) XAR membership function (B) XAM membership functions (C) YVR membership functions (D) ZLR membership functions (E) XASI membership function (F) XAST membership function

The system contains two secret membership functions with five linguistic terms with names such as “Pcond” and “Scond.” Each of these secret functions contains five terms accepted as the main rule block’s inputs. Thus, the main rule block contains 25 rules, each containing five different answers pointing to the subject’s gait ability. All rule blocks were constructed manually. Physical rehabilitation experts checked the all rules and evaluate the results one by one for each of them. Brevity prevent us from publishing full of first rule block in Table 1. Table 2 shows symmetri evaluation rules while Table 3 shows main rules. The system output membership function contains five linguistic terms as healthy, classification stages of hemiplegic gate. Figure 7 (a), (b), (c), and (d) show input membership functions.

Table 1. Some part of signal amplitudes rules

IF				THEN
XAM	**XAR**	**YVR**	**ZLR**	**Pcond**
low	low	low	low	Verybad
low	low	low	medium	Verybad
low	low	low	high	Bad
low	low	medium	low	Verybad
low	low	medium	medium	Bad
low	low	medium	high	Notgood
low	low	high	low	Bad
low	low	high	medium	Notgood

Table 2. Symmetri evaluation rules

IF		THEN
XASI	**XAST**	**Scond**
low	low	Good
low	medium	Notgood
low	high	Bad
medium	low	Notgood
medium	medium	Bad
medium	high	Verybad
high	low	Bad
high	medium	Verybad
high	high	Worst

Table 3. Main rules

IF		THEN
Pcond	**Scond**	**Classification**
Good	Good	Healthy
Good	Notgood	ST6
Good	Bad	ST6
Good	Verybad	ST5
Good	Worst	ST4
Notgood	Good	Healthy
Notgood	Notgood	ST6
Notgood	Bad	ST6
Notgood	Verybad	ST5
Notgood	Worst	ST4
Bad	Good	ST6
Bad	Notgood	ST6
Bad	Bad	ST5
Bad	Verybad	ST4
Bad	Worst	ST4
Verybad	Good	ST6
Verybad	Notgood	ST5
Verybad	Bad	ST4
Verybad	Verybad	ST4
Verybad	Worst	ST3
Worst	Good	ST5
Worst	Notgood	ST5
Worst	Bad	ST4
Worst	Verybad	ST3
Worst	Worst	ST3

The prepared software we used calculates the inference in two steps: input aggregation and degree of support composition. Aggregation uses fuzzy logic operator to calculate the 'if' part's result of a production rule when the rule receives more than one input conditions. One of the linguistic conjunctions, 'and' or 'or,' link multiple input conditions. Composition links the entire condition's validity with the degree of support. The developed software uses a composition product operator to

calculate the rule's result. The composition uses the fuzzy logic operators to link the input condition to the output condition. Standard max-min and max-dot inference methods dictate that the rule's consequence is equally true as the condition. Fuzzy reasoning or fuzzy algorithm is developed to implement fuzzy implication relations. We used Tsukamoto-type fuzzy reasoning. This reasoning, proposed by Tsukamoto [22], requires that fuzzy set's membership functions are monotonic. In order to illustrate Tsukamoto-type fuzzy reasoning, two fuzzy rules from first rule block below are used:

$\Re_x$: if XAM is A_x and XAR is B_x and YVR is C_x and ZLR is D_x then $PCOND$ is E_x

……

$\Re_1$: if XAM is low and XAR is low and YVR is low and ZLR is low then $PCOND$ is $verybad$

$\Re_2$: if XAM is low and XAR is low and YVR is low and ZLR is $medium$ then $PCOND$ is $verybad$

……

$\Re_n$: if XAM is A_n and XAR is B_n and YVR is C_n and ZLR is D_n then $PCOND$ is E_n

fact : XAM is $\overline{XAM}_0$ and XAR is $\overline{XAR}_0$ and YVR is $\overline{YVR}_0$ and ZLR is $\overline{ZLR}_0$

Using Tsukamoto's method, assuming that Z_1 is the first rule of the firing degree where $W_1=C_1$ (Z_1) and the Z_2 is the second rule of the firing degree where $W_2=C_2$ (Z_2), we expressed a crisp output as weighted average. Equation 1.shows the crisp output.

$$Z = \frac{W_1 \cdot Z_1 + W_2 \cdot Z_2}{W_1 + W_2} \tag{1}$$

4 Results and Discussion

Fuzzy logic brings new possibilities into control, modeling, data analysis, diagnostics, decision-making, and other applied fields in the biomedical sciences. We used fuzzy logic classification system first to discriminate healthy subjects from patients rather than classifying those using Brunnstrom stages. We measured the acceleration signals using the accelerometry technique during walking. Developed fuzzy systems were previously tested with training data and blind approach data. Simulation studies must be conducted to validate the classification results.

We used correlation analysis to measure the association degree between the two data sets that are recorded manually for evaluation results and Fuzzy Logic Classification (FCA). The Pearson product moment correlation coefficient, r, measures the strength of the linear relationship between two variables. The mean correlation coefficient was 0.85 ($P<0.001$) for the entire cohort.

Thirty-three patients were used for these statistics. Coefficient of determination value, r2, was 0.72. We used SPSS 12.0 for Windows (Apache Software Foundation) for all calculations.

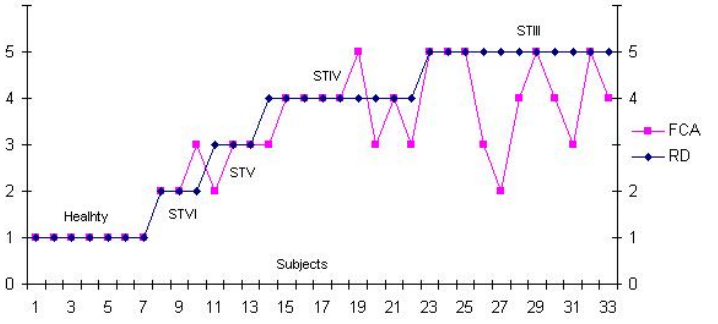


Fig. 8. FCA results comparison with Real Data (RD)

Figure 8 compares the results obtained by using random patient files. Our simulation test results showed that proposed system classify correctly 100% of subjects as patient and healthy elderly. The main problem for system is to classify post stroke paitents to correct Brunnstrom stages. Figure 8 clearly shows the problem. The difficulty is getting bigger trough to STIII. Just 5 of 11 patients from STIII were succesfully classified. 3 of 11 STIII patients were classified STIV patient while 2 of 11were classified STV and 1 of 11 was classified STVI. The last one is also the worst result for this system. When we examined the data of that patient, we realised that the patient data was very smiliar to STVI data. Actually the problem is here is strongly related to our inputs to system. In this point we can say we need to find new signal features for gait analysis to correctly discriminate to patients to Brunnstrom stages. One of our future work is feature exraction of gait signal again.

Beside the above mentioned, this study shows the possibility of using fuzzy logic to discriminate healthy subjects or patients. Successful classification of patients using Brunnstrom stages, because of unstable signal behavior, is extremely difficult. Classifying patients becomes increasingly difficult linearly according to hemiplegia's severity. Our results suggest that combination of artificial intelligence techniques may be suitable to assess the hemiplegic gate. Besides fuzzy systems, neural nets and adaptive neuro-fuzzy inference systems can be used for effective physical therapy, our current research interest.

Acknowledgment. The authors specially thanks to Dr.Toshiyo Tamura and Dr.Masaki Sekine for their great help to study during sabbatical stay in Chiba-Japan, and Akdeniz University Scientific Research Projects Unit for financial support.

References

1. Baum, H.M., Robins, M.: The national survey of stroke. Survival and prevalence. Stroke 12(2), 159–168 (1981)
2. Morita, S., Yamamoto, H., Furuya, K.: Gait analysis of hemiplegic patients by measurement of ground reaction force. Scand. J. Rehabil. Med. 27(1), 37–42 (1995)
3. Morris, J.R.W.: Accelerometry-A technique for the measurement of human body movements. J. Biomechanics 6(6), 467–471 (1981)

4. L.A., Z.: Fuzzy Sets. Information and Control 8, 338–353 (1965)
5. Andriacchi, T.P.: Clinical applications of gait analysis. In: Proceedings of the 2nd North American Congress on Biomechanics. Proc. NACOB II, NACOB, Chicago, IL (1992)
6. Benedetti, M.G., Catani, F., Leardini, A, Pignotti, E, Giannini, S: Data management in gait analysis for clinical applications. Clin. Bio. mech. 13, 204–215 (1998)
7. Dubois, D., Prade, H.: An introduction to fuzzy systems. Clin. Chim. Acta 270, 3–29 (1998)
8. Sekine, M., Tamura, T., Akay, M., Fujimoto, T., Togawa, T., Fukui, Y.: Discrimination of Walking Patterns using Wavelet-Based Fractal Analysis. IEEE Transactions on Neural Systems and Rehabilitation Engineering 10(3), 188–196 (2002)
9. Dubois, D., Prade, H.: An introduction to fuzzy systems. Clin. Chim. Acta. 270, 3–29 (1998)
10. Kuncheva, L.I., Steimann, F.: Fuzzy diagnosis. Artif. Intell. Med. 16, 121–128 (1999)
11. Nauck, D., Kruse, R.: Obtaining interpretable fuzzy classification rules from medical data. Artif. Intell. Med. 16, 149–169 (1999)
12. Usher, J., Campbell, D., Vohra, J., Cameron, J.: A fuzzy logic-controlled classifier for use in implantable cardioverter defibrillators. Pace-Pacing Clin Electrophysiol 22, 183–186 (1999)
13. Belal, S.Y., Taktak, A.F.G., Nevill, A.J., Spencer, S.A., Roden, D., Bevan, S.: Automatic detection of distorted plethysmogram pulses in neonates and paediatric patients using an adaptive-network-based fuzzy inference system. Artif. Intell. Med. 24, 149–165 (2002)
14. Guler, I., Ubeyli, E.D.: Application of adaptive neuro-fuzzy inference system for detection of electrocardiographic changes in patients with partial epilepsy using feature extraction. Expert Syst. Appl. 27(3), 323–330 (2004)
15. Ubeyli, E.D., Guler, I.: Automatic detection of erythemato-squamous diseases using adaptive neuro-fuzzy inference systems. Comput. Biol. Med. 35(5), 421–433 (2005)
16. Ubeyli, E.D., Guler, I.: Adaptive neuro-fuzzy inference systems for analysis of internal carotid arterial Doppler signals. Comput. Biol. Med. 35, 687–702 (2005)
17. Bussmann, J.B.J., Damen, L., Stam, H.J.: Analysis and decomposition of signals obtained by thigh-fixed uni-axial accelerometry during normal walking. Med. Biol. Eng. Comput. 38, 632–638 (2000)
18. Evans, A.L., Duncan, G., Gilchrist, W.: Recording accelerations in body movements. Med.& Biol. Eng. & Comput. 29, 102–104 (1991)
19. Sadeghi, H., Allard, P., Prince, F., Labelle, H.: Symmetry and Limb dominance in able-bodied gait:a review. Gait and Posture 12, 34–45 (2000)
20. Aminian, K., Rezahhanlou, K., Andres, E., Fritsch, C., Leyvraz, P.F., Robert, P.: Temporal feature estimation during walking using miniature accelerometers: an analysis of gait improvement after hip arthroplasty. Medical & Biological Engineering & Computing 37, 686–691 (1999)
21. Robinson, R.O., Herzog, W., Nigg, B.M.: Use of force platform variables to quantify the effects of chiropractic manipulation on gait symmetry. J. Manipulative Physiol. Ther. 10, 172–176 (1987)
22. Tsukamoto, Y.: An Approach to Fuzzy Reasoning Model. In: Advances in fuzzy Set Theory and Applications, North-Holland, Amsterdam (1979)

Traffic Sign Recognition Using Discriminative Local Features

Andrzej Ruta, Yongmin Li, and Xiaohui Liu

School of Information Systems, Computing and Mathematics
Brunel University, Uxbridge, Middlesex UB8 3PH, UK
{Andrzej.Ruta,Yongmin.Li,Xiaohui.Liu}@brunel.ac.uk

Abstract. Real-time road sign recognition has been of great interest for many years. This problem is often addressed in a two-stage procedure involving detection and classification. In this paper a novel approach to sign representation and classification is proposed. In many previous studies focus was put on deriving a set of discriminative features from a large amount of training data using global feature selection techniques e.g. Principal Component Analysis or AdaBoost. In our method we have chosen a simple yet robust image representation built on top of the Colour Distance Transform (CDT). Based on this representation, we introduce a feature selection algorithm which captures a variable-size set of local image regions ensuring maximum dissimilarity between each individual sign and all other signs. Experiments have shown that the discriminative local features extracted from the template sign images enable simple minimum-distance classification with error rate not exceeding 7%.

1 Introduction

Recognition of traffic signs has been a challenge problem for many years and is an important task for the intelligent vehicles. Although the first work in this area can be traced back to the late 1960's, only in the 1990's, when the idea of autonomous intelligent navigation was popularised, significant advances were made. Nevertheless, there is still an apparent gap between what human and machine can do, making the attentive driver an irreplaceable guarantor of safety in the traffic environment.

Road signs have unique properties distinguishing them from the multitude of other outdoor objects. These properties were benefited from in numerous attempts to build an efficient detection and recognition system. In the majority of published work a two-stage sequential approach was adopted, aiming at locating the regions of interest first, and subsequently passing them to the classifier [1,2,3]. To detect possible sign candidates traditionally colour information is extracted [1,2], followed by the geometrical edge [1,4] or corner analysis [2]. Alternative approaches utilise distance transform [5] or neural networks [6]. In several studies the geometrical tracking aspect was given consideration [1,6,7]. However, reliable

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 355–366, 2007.

prediction of the geometrical properties of signs from a moving vehicle is complex in general as the vehicle's manoeuvres are enforced by the actual traffic situation and therefore cannot be apriori known. To overcome this problem in the absence of on-line external sensor measurements, the above approaches impose simplified motion model, e.g. assuming constant velocity. In the classification stage a pixel-based approach is often adopted and the class of the detected sign is determined by the cross-correlation template matching [1] or neural network [2]. Feature-based approach is used for instance in [3]. More recently, Bahlmann et al. [9] adopted the ideas of Viola and Jones [8] to detect traffic signs based on the colour-sensitive Haar wavelet features and AdaBoost framework. In the classification stage, assuming Gaussian class distribution and the independence of consecutive frame observations, Bayes classifier is used to fuse the individual observations over time. Only 6% error rate is reported using this method. Paclík et al. [10] introduced a different strategy built upon the claim that a candidate sign can be represented as a set of similarities to the stored prototype images. For each class similarity assessment is made with respect to a different set of local regions refined in the training process.

In this work we have developed a two-stage road sign detection and classification system. Figure 1 shows an example frame from video input with a road sign detected and recognised. More specifically, our detector is a form of well-constrained circle/regular polygon detector introduced in [4], augmented with the appropriate colour pre-filtering. In the classification stage, motivated by [10], we introduce a novel feature selection algorithm built on top of the Colour Distance Transform (CDT) image representation. We show that although our algorithm generates sign descriptors of variable dimensionality, individual classification scores can be made directly comparable due to the global selection criterion used. In consequence the proposed method seems to be a more natural way of discrimination among signs, as not the same amount of information is necessary to tell different classes apart. The rest of this paper is organised as follows: In section 2 traffic sign detection and tracking are briefly described. Sections 3 and 4 discuss the main contributions of this work, discriminative region selection and sign classification. Section 5 presents experimental results on the real traffic video sequences. Finally, conclusions are drawn in section 6.

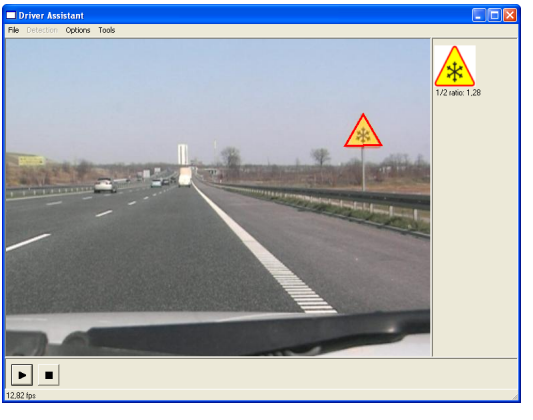


Fig. 1. Screenshot from our traffic sign recognition system in action

2 Sign Detection and Tracking

Our road sign detector is triggered every fixed number of frames to capture new candidates emerging in the scene. It makes use of the apriori knowledge about the model signs, which are uniquely identified in terms of the general shape, colour and contained ideogram. Based on the first two properties four sign categories corresponding to the well-known semantic families are identified: instruction (blue circular), prohibitive (red circular), cautionary (yellow triangular), and informative (blue square) signs. As we believe the shape of a sign and its boundary colour are sufficient visual cues to locate the candidates reliably, the proposed detector operates on the colour gradient and edge maps of the original video frames. Furthermore, it uses a generalisation of Hough Transform introduced in [4], which is motivated by the fact that targeted objects are all instances of equiangular polygons, including circles that can be though of as such polygons with the infinite number of sides. Original regular polygon transform is augmented with the appropriate image preprocessing intended to enhance edges of specific colour. For each RGB pixel $\boldsymbol{x} = [x_R, x_G, x_B]$ and $s = x_R + x_G + x_B$, a simple colour enhancement is provided by a set of transformations:

$$\begin{aligned} f_R(\boldsymbol{x}) &= \max(0, \min((x_R - x_G)/s, (x_R - x_B)/s)) \\ f_B(\boldsymbol{x}) &= \max(0, \min((x_B - x_R)/s, (x_B - x_G)/s)) \\ f_Y(\boldsymbol{x}) &= \max(0, \min((x_R - x_B)/s, (x_G - x_B)/s)) \end{aligned} \quad . \tag{1}$$

First two transforms extract these parts of the image where the red or blue component respectively dominate the most over both remaining components. The third formula has similar meaning, but as the pure yellow colour has equal value in the red and green channels and zero in blue channel, it attempts to enhance pixels where both former components dominate the most over the latter.

In the resulting images red, blue, and yellow edge maps are extracted by a simple filter which for a given pixel picks the highest difference among the pairs of neighbouring pixels that could be used to form a straight line through the middle pixel being tested. Obtained values are further thresholded and only in the resulting edge pixels values of directional and magnitude gradient are calculated. This technique is adequate to our problem as it enables quick extraction of edges and avoids expensive computation of the whole gradient magnitude map which, with the exception of the sparse edge pixels, is of no use to the shape detector. For a given pair of gradient and edge images associated with colour c, circle and regular polygon detectors yield a number of possible sign shapes. This number depends on the actually set detector's sensitivity defined by a fixed, relatively low threshold value specifying a percentage of the maximum possible number of votes which would be accumulated in presence of the regular shape of known radius and ideally sharp gradient along the contour. As each candidate has known shape (either circle, or equilateral triangle, or square), and border colour c, detector serves as a pre-classifier reducing the number of possible templates to analyse to the ones contained in either category. When signs are in the cluttered background, a number of false candidates may be produced. To address

this problem, an additional step is taken to verify the presence of sign-interior colours appropriate for the just found category.

Once a candidate sign is detected, exhaustive search in consecutive frames is unnecessary. Assuming motion with constant velocity along the optical axis of the camera, we have employed a Kalman filter [11] to track a sign detected in a previous frame of an input video. The state of the tracker is defined by (x, y, s), where x, y are coordinates of the sign's centre in the image, and s is the scale factor to the standard sign templates. In the current implementation we use the mean and variance estimates from the Kalman filter to locate the centroid and the size of the local search region in the next frame. Therefore, computation has been significantly reduced compared to exhaustive search over the whole image.

3 Image Representation and Feature Selection

Selecting an optimal feature set for a large number of template images is a non-trivial task. We have experimented with several techniques such as Principal Component Analysis and AdaBoost. Aiming at retrieving the global variance of a whole data set, PCA is not capable of capturing features critical to the individual templates. AdaBoost on the other hand, although generating efficient classifiers, is not entirely convincing in terms of the fixed cardinality of the feature set being extracted. Clearly, certain signs are very distinctive and analysis of only a few small regions enables distinguishing them even among tens of others. Meanwhile, there are groups of very similar signs that look tightly clustered, even in a highly multidimensional feature space. This complex nature of similarity between templates raises a question whether there is sufficient justification for classifying signs based on the same set of features.

Motivated by [10], we propose here an algorithm that relaxes the above limitation by extracting for each model sign a limited number of local image regions in which it looks possibly the most different from all other templates residing in the same category. The same discriminative regions are further used to compare a video frame image with the templates and make a reliable on-line classification. Below we first outline the process of converting the raw bitmap images into a more suitable discrete-colour representation. Second, we introduce the notion of local image region and dissimilarity. Finally, the implementation of the discriminative region selection algorithm is formalised and discussed.

3.1 Colour Discretisation

Detected sign images come as rectangular regions containing the target object and, depending on its shape, also background fragments, as depicted in Fig. 2. In order to prepare the candidate for classification, the image is first scaled to a common size, typically 60×60 pixels. Undesirable background regions are then masked out using the information about the object's shape and orientation provided by the detector [4]. It is important to note that the full colour spectrum is far more than necessary to identify the object, as the template signs contain

only up to four distinctive colours per category. Therefore, the candidate images are finally subject to colour discretisation as follows:

1. For the template sign images, which already contain ideal colours, discretisation merely aims at changing the physical representation from 24-bit RGB bitmap to 2-bit image with the specific colour indices encoded. A set of thresholds similar to these used in [12] is applied to the templates in Hue-Saturation-Value space to complete this task.
2. For on-line colour segmentation category-specific colours are learned from a set of training images taken from the real video sequences. Expectation Maximisation algorithm [13] is employed to estimate an optimal Gaussian Mixture model for each colour. The procedure is restarted several times for the increasing number of randomly initialised Gaussian components to refine the estimation. The best model in terms of mean data likelihood is selected. In order to speed up the on-line segmentation, off-line learned models are used by a Bayes classifier to assign the appropriate colour to each possible RGB triple, yielding a look-up table with 255^3 entries for each sign category. Sample results of on-line colour discretisation are illustrated in Fig. 2.

Fig. 2. Sample images obtained by sign detector before (above) and after (below) background masking and colour discretisation; 2 bits encode colours in each image

3.2 Discriminative Local Regions

The space of regions is obtained from the colour-segmented template sign images. First, for each discrete colour present in the image a separate distance transform [14] is computed, producing images similar to these shown in Fig. 3. In DT computation pixels of given colour are simply treated as feature pixels and all the remaining ones as non-feature pixels. $(3, 4)$ Chamfer metric [15] is used to approximate Euclidean distance between the feature pixels. To emphasise the strong relation to colour, we call this variant of DT a Colour Distance Transform (CDT). In the next step image is divided into 4×4-pixel regions. Within each region r_k local dissimilarity between the images I and J can be calculated using discrete colour image of I and CDT images of J by averaging pixel-wise distances:

$$d_{r_k}(I, J) = \frac{1}{m} \sum_{t=1}^{m} d_{CDT}(I(p_t), J(p_t)) \ , \tag{2}$$

where for each of m pixels p_t contained in the region, distance $d_{CDT}(I(p_t), J(p_t))$ is picked from the appropriate CDT image of J, depending on the colour of this

pixel in I. Let us also denote by $\widehat{d_{\mathbf{S}}}(I,J)$ and $\widehat{d_{\mathbf{S},\mathbf{W}}}(I,J)$ a normal and weighted average local dissimilarities between the images I and J computed over regions $r_k \in \mathbf{S}$ (weighted by $w_k \in \mathbf{W}$):

$$\widehat{d_{\mathbf{S}}}(I,J) = \frac{1}{M}\sum_{k=1}^{M} d_{r_k}(I,J) \ , \tag{3}$$

$$\widehat{d_{\mathbf{S},\mathbf{W}}}(I,J) = \frac{\sum_{k=1}^{M} w_k d_{r_k}(I,J)}{\sum_{k=1}^{M} w_k} \ . \tag{4}$$

Obviously, as distance transforms for template signs are pre-computed, on-line comparison between the corresponding regions of the detected candidate image and model sign images can run extremely fast.

(a) (b) (c) (d)

Fig. 3. Colour distance transform images: original discrete colour image (a), black CDT (b), white CDT (c), red CDT (d); darker regions denote shorter distance

3.3 Region Selection Algorithm

Assuming pre-determined category of signs $C = \{T_i : i = 1,\ldots,N\}$ and a candidate image x_j, our goal is to determine the class of x_j by maximising posterior:

$$p(T_i|x_j,\theta_{\mathbf{i}}) = \frac{p(x_j|T_i,\theta_{\mathbf{i}})p(T_i)}{\sum_{i=1}^{N} p(x_j|T_i,\theta_{\mathbf{i}})} \ . \tag{5}$$

Our objection to using a uniform feature space for classification makes us envisage different model parameters $\theta_{\mathbf{i}} = (\mathbf{I_i}, \mathbf{W_i})$ for each template T_i. $\mathbf{I_i}$ denotes an indexing variable determining the set $\mathbf{S_i}$ of regions to be used and $\mathbf{W_i}$ is a vector of relevance corresponding to the regions $r_k \in \mathbf{S_i}$ selected by $\mathbf{I_i}$. In order to learn the best model parameters $\theta^*_{\mathbf{i}}$, the following objective function is maximised:

$$O(\theta_{\mathbf{i}}) = \sum_{j \neq i} \widehat{d_{\mathbf{S_i}}}(T_j,T_i) \ . \tag{6}$$

In other words, the regions best characterising a given sign are obtained through maximisation of the sum of local dissimilarities between this sign's template and all the remaining signs' templates. In presence of model images only, each term $\widehat{d_{\mathbf{S_i}}}(T_j,T_i)$ as a function of the number of discriminative regions is necessarily monotonically decreasing. As a result, there would always be just a single best region or a few equally good regions maximising (6). In practice, such sign descriptors are unlikely to work well for video frames, typically affected by a severe noise, where more support in terms of the number of image patches to match

is required to make a reliable discrimination. Our objective function, as described below, is hence only optimised up to the specified breakpoint, yielding a representation which is richer and thus more trustworthy in a real-data scenario.

1. **input:** sign category $C = \{T_j : j = 1, \ldots, N\}$, target template index i, region pool $R = \{r_k : k = 1, \ldots, M\}$, dissimilarity threshold t_d
2. **output:** target set F_i of regions with associated weights
3. initialise an array of region weights $W = \{w_k : w_k = 0, k = 1, \ldots, M\}$
4. for each template $T_j \in C$, $j \neq i$ do
   (a) find region $r_j^{(1)}$ such that $d_{r_j^{(1)}}(T_j, T_i) = \max_k d_{r_k}(T_j, T_i)$
   (b) initialise ordered region list $F_j = [r_j^{(1)}]$ storing the selected regions to discriminate between templates T_i and T_j
   (c) initialise remaining feature pool $P_j = R \setminus \{r_j^{(1)}\}$
   (d) initialise region counter $l = 1$
   (e) while not STOP do
      i. increment region counter $l = l + 1$
      ii. for each region $r_k \in P_j$ construct a region list $S_k = F_j + r_k$ and pick region $r_j^{(l)}$ maximising $\widehat{d_{S_k}}(T_j, T_i)$
      iii. insert region $r_j^{(l)}$ at the beginning of the selected region list $F_j = r_j^{(l)} + F_j$
      iv. update the remaining region pool $P_j = P_j \setminus \{r_j^{(l)}\}$
      v. STOP if $\widehat{d_{F_j}}(T_j, T_i) < t_d d_{r_j^{(1)}}(T_j, T_i)$
   (f) update found region weights $w_k = w_k + p_k$ for all regions $r_k \in F_j$, where p_k denotes rank (position in the list) of the k-th region
5. build target region set $F_i = \{(r_k, w_k) : w_k > 0\}$

Similarly to Paclík et al. [10], in the model training stage we have adopted elements of a sequential forward search strategy, a greedy technique from the family of floating search methods [16]. However, both approaches differ significantly in the two main aspects. First, we think that learning the signs from the real-life images is counter-intuitive as the publicly available templates characterise the respective classes fully. Second, we believe that the possible within-class appearance variability may well be accounted for by a robust distance metric, as the one introduced in (2-4), instead of being learned. Our implementation then picks a given template sign and compares it to each of the remaining templates. In each of such comparisons the algorithm loops until the appropriate number of local regions are found. It should be noted that at a given step of the loop the most dissimilar region is fixed and removed from the pool of available regions. Moreover, at the k-th step the distance between the considered image and the image being compared to is measured with respect to the joint set comprised of the new k-th region and all previously found regions. At the end of the loop

an ordered list of regions is produced, sorted by their decreasing discriminative power. Each pairwise region set build-up is controlled by a global threshold, t_d, specifying the minimum allowed dissimilarity between any pair of templates being compared as a percentage of the maximum possible dissimilarity, i.e. the one for just a single most discriminative region. Such a definition of STOP criterion ensures that the same amount of dissimilarity between any pair of templates is incorporated in the model. This in turn allows us to treat different sign classes as directly comparable, irrespective of their actual representation. The final set for each class is constructed by merging the pair-specific subsets which is reflected in the region weights carrying the information on how often and with what rank each particular region was selected.

For each sign the above algorithm yields a set of its most unique regions. It should be noted that in the final step, depending on the actual dissimilarity threshold specified, certain number of regions will be found completely unused, and hence discarded. An example of our feature selector's output is depicted in Fig. 4. Obtained discriminative region maps clearly show that different signs are best distinguishable in different fragments of the contained pictogram. It can also be seen that although the same value of global parameter t_d was used, different numbers of meaningful regions remained.

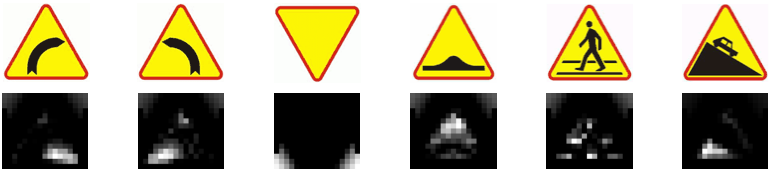


Fig. 4. Sample triangular template images (above), and discriminative regions obtained for parameter $t_d = 0.7$ (below); brighter regions correspond to higher dissimilarity

4 Temporal Classifier Design

A road sign classifier distinguishes between the sign classes contained in a category pre-determined in the detection stage, based on the discriminative feature representation unique for each particular sign. For simplicity two assumptions are made: 1) the dissimilarity between each sign and all other same-category signs is Gaussian-distributed in each local region and independent of the dissimilarities in all other regions characterising this sign, and 2) class priors $P(T_i)$ are equal. In such a case Maximum Likelihood theory allows us to relate the maximisation of likelihood $p(x_j|T_i, \theta_{\mathbf{i}})$ to the minimisation of distance $\widehat{d_{\mathbf{S_i},\mathbf{W_i}}}(x_j, T_i)$ over i. Therefore, for a known category $C = \{T_i : i = 1, \ldots, N\}$, and observed candidate x_t at time t, the winning class $L(x_t)$ is determined from (5):

$$L(x_t) = \arg\max_i p(x_t|T_i, \theta_{\mathbf{i}}) = \arg\min_i \widehat{d_{\mathbf{S_i},\mathbf{W_i}}}(x_t, T_i) \ , \tag{7}$$

where the elements of the region set $\mathbf{S_i}$ and the corresponding weights in $\mathbf{W_i}$ denote the ones learned in the training stage for template T_i.

When a series of observations from a video sequence is available, it is reasonable to integrate the classification results through the whole sequence over time, instead of performing individual classifications. Hence, at a given time point t our temporal integration scheme attempts to incorporate all the observations made since the sign was for the first time detected until t. Denoting observation relevance by $q(t)$ and assuming independence of the observations from consecutive frames, the classifier's decision is determined by:

$$L(X_t) = \arg\min_i \sum_{k=1}^{t} q(t) \widehat{d_{S_i,W_i}}(x_k, T_i) \ . \tag{8}$$

We have observed that the signs detected in the early frames are inaccurate and contain blended pictograms due to the low resolution. Also as colours tend to be paler when seen from the distance, previously discussed colour discretisation exposes severe limitations, unless performed for later frames depicting candidate sign already grown in size. To address this problem, we adopt the exponential observation weighting scheme from [9] in which relevance $q(t)$ of observation x_t depends on the candidate's age (and thus size):

$$q(t) = b^{t_0 - t} \ , \tag{9}$$

where $b \in (0, 1]$ and t_0 is the time point when the sign is for the last time seen.

5 Experiments

To evaluate our traffic sign recognition system, experiments were performed on the real data collected on Polish roads. Sample video sequences were acquired from a moving car with a DV camcorder mounted firmly in front of the windscreen, and subsequently divided into short clips for off-line testing. Video content depicts the total of 144 signs and includes urban, countryside, and motorway scenes in natural lightning during daytime, with numerous signs appearing in shade and in cluttered background. Table 1 illustrates obtained results.

As seen in Tab. 1, obtained real-time classification error rate does not exceed 7%, making our method comparable to the recently published ones [9,10]. However, it should be noted that our template database contains significantly more signs than in any of the previous studies. Direct comparison with the respective algorithms is not possible as neither the test data nor the details of its acquisition are made available. Repetitions of the experiment for different values of dissimilarity threshold confirmed our expectations. For each category of signs different thresholding provides the best results, which depends not only on the category size, but primarily on the diversity of ideograms in the contained signs. The following observation is vital at this point. The optimal threshold for each category must strike a balance between the two: maximising template signs' separability and the reliability of the obtained dissimilarity information in the real

Table 1. Recognition performance for different values of dissimilarity threshold t_d and temporal weight base $b = 0.8$; number of classes in each category: red circles (RC), blue circles (BC), yellow triangles (YT), and blue squares (BS) is given in parentheses and the best classification rate is highlighted

	t_d	RC (55)	BC (25)	YT (42)	BS (13)	overall (135)
detected	–	86.4%	100.0%	96.3%	94.4%	95.8%
recognised	0.9	100.0%	90.6%	73.6%	91.2%	85.5%
	0.7	100.0%	100.0%	88.7%	70.6%	87.7%
	0.5	89.5%	96.9%	79.2%	58.3%	80.4%
	best	100.0%	100.0%	88.7%	91.2%	93.5%

data context. Very high threshold values lead to separation of a very few good regions for a particular model sign, however such sparse information may not be sufficiently stable to classify correctly a possibly distorted, blurred, or occluded object in a video frame. Very low threshold values on the other hand introduce information redundancy by allowing image regions that contribute little to the uniqueness of a given sign. In a resulting feature space signs are simply more similar to one another and hence more difficult to tell apart.

In terms of the detection, most of failures were caused by the insufficient contrast between a sign's boundary and the background, especially for pale-coloured and shady signs. In a few cases this low contrast was caused by the poor quality of the physical target objects rather than their temporarily confusing appearance. Single detection errors emerged when two signs were mounted closely on one pole. In this particular situation candidate objects may be confused with each other, as the local search region of one candidate always contains at least part of its neighbour. Detection proved to be the computationally most expensive part of the system, however processing speed of the entire algorithm including classification is 10-20 fps on a standard PC, depending on the actual difficulty of the scene.

After closer investigation we observed that approximately one in three classification errors resulted from confusion between the nearly identical classes, e.g. pedestrian crossing and bicycle crossing. Differences between such signs were found difficult to capture, resulting sometimes in the correct template receiving the second best score. Colour segmentation appeared to be resilient to variations of illumination, leading directly to failure in only a few cases when the signs were located in a very shady area or were themselves of poor quality. This can be a proof of usefulness of Gaussian Mixture colour modelling. Remaining failures can be attributed to the limitations of the detector. Although distance transform, utilised in dissimilarity computation, and observation weighting neutralise inaccurate detection effects to a large extent, they are of little help in presence of certain phenomena consistent in their nature. Two examples of such situations are remarkable:

1. Some signs' ideograms consist of edges that may actually be easier to detect than the boundary. This may cause detected shape to appear clipped.

2. Signs very close to the camera being distorted in perspective projection usually receive sufficient score in the detector's accumulator space. These signs are yet still detected as regular shapes, resulting in the inaccurate shape estimation.

As indicated in the previous section, in the early video frames, due to the low resolution, delimitation of sign's contour and subsequent colour discretisation are usually less accurate. Frequently the correct decision can be developed by the classifier from just the last several frames where the sign's shape is the most stably detected and its content the most clearly visible. This fact has severe implications on the candidate classification, which is a good justification for our exponential observation weighting used to promote the most recent measurements. Apparently, the classification accuracy with weighting enabled is by 10-20 % higher, depending on the weight base b used.

6 Conclusions

In this paper we have introduced a novel image representation and discriminative feature selection method for road sign recognition where a large number of classes are involved. The proposed algorithms have been tested on the real video sequences, yielding a low classification error rate. It was shown that on top of a Colour Distance Transform (CDT) representation highly discriminative sign descriptors can be extracted based on the principle of dissimilarity maximisation. With these descriptors available, a conventional classifier is able to compete with the state-of-the-art sign recognition systems, operating in close to real time. In comparison to the previous studies, our method seems attractive in three aspects. First, feature selection is performed directly on the publicly available template sign images. Second, each template is treated on an individual basis which is reflected in the number, position, and importance of the local image regions extracted in order to achieve a desired level of dissimilarity from the remaining templates. Finally, by using a Colour Distance Transform (CDT) we have shown that the proposed dissimilarity-based description of signs can well be extended from model images to the real video frames as the resulting distance measure is made smoother and thus more resistant to various types of noise typically affecting the video content.

References

1. Piccioli, G., De Micheli, E., Parodi, P., Campani, M.: A robust method for road sign detection and recognition. Image and Vision Computing 14(3), 209–223 (1996)
2. de la Escalera, A., Moreno, L.E., Salichs, M.A., Armingol, J.M.: Road traffic sign detection and classification. IEEE Trans. on Industrial Electronics 44(6), 848–859 (1997)
3. Paclík, P., Novovicova, J., Pudil, P., Somol, P.: Road Sign Classification using the Laplace Kernel Classifer. Pattern Recognition Letters 21(13-14), 1165–1173 (2000)

4. Loy, G., Barnes, N., Shaw, D., Robles-Kelly, A.: Regular Polygon Detection. In Proc. of the 10th IEEE Int. Conf. on Computer Vision 1, 778–785 (2005)
5. Gavrila, D.: Multi-feature Hierarchical Template Matching Using Distance Transforms. In: Proc. of the IEEE Int. Conf. on Pattern Recognition, Brisbane, Australia, pp. 439–444. IEEE, Los Alamitos (1998)
6. Fang, C.-Y., Chen, S.-W., Fuh, C.-S.: Road-Sign Detection and Tracking. IEEE Trans. on Vehicular Technology 52(5), 1329–1341 (2003)
7. Miura, J., Kanda, T., Shirai, Y.: An active vision system for real-time traffic sign recognition. In: Proc. of the IEEE Conf. on Intelligent Transportation Systems, Darborn, MI, USA, pp. 52–57. IEEE, Los Alamitos (2000)
8. Viola, P., Jones, M.: Robust Real-time Object Detection. International Journal of Computer Vision 57(2), 137–154 (2004)
9. Bahlmann, C., Zhu, Y., Ramesh, V., Pellkofer, M., Koehler, T.: A System for Traffic Sign Detection, Tracking and Recognition Using Color, Shape, and Motion Information. In: Proc. of the IEEE Intelligent Vehicles Symposium, pp. 255–260 (2005)
10. Paclík, P., Novovicová, J., Duin, R.P.W.: Building Road-Sign Classifiers Using a Trainable Similarity Measure. IEEE Trans. on Intelligent Transportation Systems 7(3), 309–321 (2006)
11. Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. Trans. of the ASME - Journal of Basic Engineering 82, 35–45 (1960)
12. Zhang, L., Lin, F., Zhang, B.: A CBIR method based on color-spatial feature. In: Proc. of the IEEE Region 10 Conf. TENCON 99, vol. 1, pp. 166–169 (1999)
13. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society (B) 39(1), 1–38 (1977)
14. Borgefors, G.: Distance transformations in digital images. Computer Vision, Graphics, and Image Processing 34(3), 344–371 (1986)
15. Akmal Butt, M., Maragos, P.: Optimum design of chamfer distance transforms. IEEE Trans. on Image Processing 7(10), 1477–1484 (1998)
16. Pudil, P., Novoviová, J., Kittler, J.: Floating search methods in feature selection. Pattern Recognition Letters 15(11), 1119–1125 (1994)

Novelty Detection in Patient Histories: Experiments with Measures Based on Text Compression

Ole Edsberg, Øystein Nytrø, and Thomas Brox Røst

Norwegian University of Science and Technology, Department of Computer and Information Science, Sem Sælands vei 7-9, NO-7491 Trondheim, Norway

Abstract. Reviewing a patient history can be very time consuming, partly because of the large number of consultation notes. Often, most of the notes contain little new information. Tools facilitating this and other tasks could be constructed if we had the ability to automatically detect the novel notes. We propose the use of measures based on text compression, as an approximation of Kolmogorov complexity, for classifying note novelty. We define four compression-based and eight other measures. We evaluate their ability to predict the presence of previously unseen diagnosis codes associated with the notes in patient histories from general practice. The best measures show promising classification ability, which, while not enough to serve alone as a clinical tool, might be useful as part of a system taking more data types into account. The best individual measure was the normalized asymmetric compression distance between the concatenated prior notes and the current note.

1 Introduction

Patient histories, as recorded in clinical information systems, contain records, typically both with structured and free-text components, of encounters between care provider and patient. Such histories can get very long, putting a great burden on those who have to review them. This is particularly true in general practice and for patients with chronic disease. Often, many entries in a history contain repetitions of already known information or reports of follow-up of established treatment of known problems. The ability to automatically pick out the novel parts of a history could have great utility in the following areas:

1. Improving review of histories by highlighting novel parts or folding non-novel parts.
2. Summarizing histories.
3. Automatically notifying collaborating personnel, keeping them up-to-date on important events.

It could also be useful in segmenting or filtering patient histories as pre-processing for data mining.

To perform optimally, a novelty detector for patient histories would need to take many types of data into account, including textual notes, diagnosis codes,

M.R. Berthold, J. Shawe-Taylor, and N. Lavrač (Eds.): IDA 2007, LNCS 4723, pp. 367–378, 2007.

lab results, prescriptions and referrals. In this work, we will nevertheless only use the textual notes, since we think that the parts of the problem should be studied separately before it is attacked as a whole. The problem can be formulated as follows: *Given the previous notes in a patient history, classify the current note as novel or not novel.* The obvious problem with this formulation is that novelty is a subjective criterion, and we need an objective gold-standard for evaluating our methods. One possibility would be to hire clinicians to read patient histories and rate the novelty of notes. This would be very costly for a sufficiently large data set, and there would still be problems with subjectivity, since different clinicians might have different concepts of novelty. Therefore, in this work we instead draw our evaluation criterion from the data. The notes in our patient histories have associated diagnosis codes indicating the reason for encounter, and we label a note as novel if one of its associated codes is given for the first time in the history. Our thinking is that a clinician who finds that the patient's situation warrants a new diagnostic category must, almost by definition, think that something novel has occurred. There might be types of novelty not captured by the evaluation criterion, such as important developments not warranting new diagnostic categories. If reflected in the textual notes, these might cause spurious false positives in the evaluation. While imperfect, our evaluation criterion gives us a basis for comparing different measures.

It may seem strange to take the evaluation criterion from information that is already available in the patient histories and could be used to trivially obtain a perfect score. However, our real goal is not to maximize the score but to examine the performance of different text-based novelty measures. As long as the criterion is a relatively realistic operationalization of novelty, high-performing text-based measures could afterwards serve as parts of a novelty detector considering more types of data and being evaluated according to a different criterion.

We have implemented twelve novelty detectors and evaluated them using the evaluation criterion introduced above. Four of them make use of string compression to approximate the absolute and conditional Kolmogorov complexity of notes and concatenations of notes. One is the baseline TF-IDF vector space method from the related problem of new event detection in new streams. Another is a support vector machine (SVM) using the eleven other detectors as input. The remaining six are primitive measures intended to serve as reality checks for the performance of the others.

Our main contributions are:

1. The formulation and formalization of the problem and evaluation framework.
2. The application of compression-based similarity measures to novelty detection in text, and in particular our proposal of the normalized asymmetric compression distance (defined in equation 7), which turned out to be the best individual performer.
3. The evaluation of the novelty detectors on a general practitioner patient record with 5346 patient histories and a total of 273991 notes to classify.

2 Related Work

We now briefly introduce the two main bodies of work our novelty detectors draw upon.

2.1 Compression-Based Similarity Measures

It is not unreasonable to assume that a document with a higher similarity to previous documents is less likely to be novel. Therefore, a similarity measure might be a useful basis for a novelty detector. Since novelty implies the presence of new information, similarity measures based on joint and separate compressibility [1,2,3,4] are intuitively appealing. The theoretical basis of these measures lies in algorithmic information theory, where $K(x)$, the Kolmogorov complexity of the bit string x, is defined as length of the shortest binary program that makes a fixed universal Turing machine output x. $K(x|y)$, the conditional Kolmogorov complexity of x given y, is the length of the shortest program that makes the machine output x when given y as input. The information distance ID between x and y is the length of the shortest program that makes the machine output x when given y and vice versa. The following has been shown to hold, up to a logarithmic, additive term [1]:

$$\mathrm{ID}(x, y) = \max\{K(x|y), K(y|x)\} \tag{1}$$

The normalized information distance between x and y, defined as

$$\mathrm{NID}(x, y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}, \tag{2}$$

has been proven to be a metric and to to be minimal (up to an additive constant) among a set of admissible distances (where the admissibility criteria include a density requirement to exclude trivial solutions) [2]. For applications, the non-computable Kolmogorov complexity can be approximated using a real-world compressor such as `gzip`, resulting in the normalized compression distance [4],

$$\mathrm{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}, \tag{3}$$

where $C(x)$ is the compressed length of x, and $C(xy)$ is the compressed length of the concatenation of x and y. (The formulation of the NCD assumes that the compressor is symmetric, i.e. that $C(xy) = C(yx)$.) The intuitive interpretation of the NCD is that two strings are similar if they can be compressed to a greater degree together than separately.

Successful applications of the NCD and its variations include clustering and phylogeny reconstruction of biological sequences [4], clustering of languages and authors represented by textual corpora [4] and clustering, classification and anomaly detection of time series [3]. The differences between Keogh et al.'s time series anomaly detection [3] and our work are 1) the type of data, 2) novelty

being defined with respect to the past history whereas anomaly is defined with respect to the whole history, 3) the types of measures used, and 4) the finding. Keogh et al. found that a variation of NCD was usful for anomaly detection. As described below, we found that the most direct application of the NCD (called `NM-NCD` below) was not very useful for novelty detection, that another application of the NCD (called `NM-MIN-NCD`) was better, and that the best individual measure (`NM-NACD`) was an asymmetric counterpart to the NCD. The applications cited above all use symmetric measures. Novelty detection is an inherently asymmetric problem, since different roles are played by the novel data and the data it is novel with respect to. Thus, unlike the case with clustering, an asymmetric measure may be more natural. The same argument could be applied to anomaly detection. Some early compression-based measures for assessing evolutionary distance between biological sequences [5,6] were asymmetric, but here we would think that asymmetry was a disadvantage since evolutionary distance is a symmetric concept (at least unless one sequence is known to be the ancestor of the other).

2.2 New Event Detection from News Streams

In the field of Information Retrieval, the problem of *New Event Detection (NED)* [7,8,9] has much in common with the problem we investigate in this article. In NED, a stream of news articles is scanned sequentially, and the task is to pick out those articles that describe an event, for example a peace treaty or a scientific discovery, for the first time. A baseline approach in NED is to represent the articles with TF-IDF (term frequency $\times$ inverse document frequency)-weighted vectors and classify an article as describing a new event if the maximum of the distances from the vector to the preceding vectors is above a given threshold. One difference between NED and our problem is that we have a separate document stream for each history and can base the TF-IDF weights on the past documents in the current history plus all documents in the other histories.

We use a measure based on TF-IDF as a point of comparison for our other methods and as a feature for our SVM. More sophisticated methods have been applied to NED, such as detection and separate consideration of different types of terms [9]. Similar approaches could be taken with patient record data, but we currently lack the necessary linguistic resources.

3 Materials and Methods

3.1 Data Source

Our data source was a patient record from a general practitioner's office. The record of a patient history is structured as a series of encounters, with notes, diagnosis codes and other associated information. For each patient, we extracted the series of encounter notes, along with the diagnosis codes associated with each encounter. We ignored histories with less than 21 encounters. The resulting data

Table 1. The ten diagnosis codes with the highest frequency of occurrence (determined with respect to the total number of code occurrences)

Code	Description	Frequency
K86	Hypertension uncomplicated	3.0%
L84	Back syndrome without radiating pain	2.4%
T90	Diabetes non-insulin dependent	2.5%
P76	Depressive disorder	2.0%
K78	Atrial fibrillation/flutter	2.0%
L93	Tennis elbow	1.9%
L92	Shoulder syndrome	1.8%
R74	Upper respiratory infection acute	1.5%
R83	Respiratory infection other	1.4%
K74	Ischaemic heart disease with angina	1.3%

set contained 5346 histories, with an average of 71 encounters per history (stddev 59), and an average of 0.79 diagnosis codes per encounter (stddev 0.65). There were 1004 different codes, and they were mostly drawn from the International Classification of Primary care (ICPC) [10]. The ten most frequent codes are shown in table 1. The 100 most frequent codes accounted for 68% of the code occurrences.

We ignored encounters where the note was blank. Also, since novelty detection is easier early in a history, we ignored the results from the first 20 encounter notes in each history. (When we did not ignore them, we got slightly better results than we report below.) After these exclusions, we had 273991 notes to classify.

3.2 Formalization of Patient History and Classification Task

We represent the history of patient p with the sequence

$$(n_1^p, D_1^p), (n_2^p, D_2^p), \ldots, (n_n^p, D_n^p),$$

where n_i^p is p's ith encounter note and D_i^p is the set of diagnosis codes associated with p's ith encounter. P denotes the set of all patients, and $len(p)$ the number of encounters in p's history.

For our evaluation criterion, the classification problem for patient p and note number $k \in [1, len(p)]$ is: given $(n_1^p, D_1^p), (n_2^p, D_2^p), \ldots, (n_{k-1}^p, D_{k-1}^p)$ and n_k^p, predict whether

$$D_k^p \setminus \bigcup_{i \in [1,k)} D_i^p = \emptyset,$$

i.e. whether the next encounter has any new diagnosis codes.

Our approach to solving the classification problem is to use a novelty measure `NM`, a function from the set of valid (p, k) pairs to $\mathbb{R}$. We classify patient p's kth note as novel if $\texttt{NM}(p, k) >= \theta$, where θ is a threshold value we can choose to achieve a desired trade-off between sensitivity and specificity. Figure 1 shows the application of a novelty measure to a history.

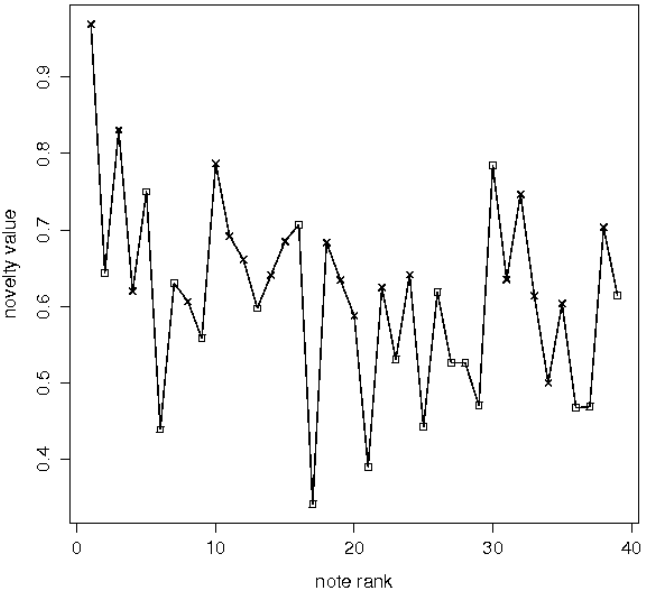


Fig. 1. The application of a novelty measure (`NM-NACD`, defined in equation 8) to every note in a history. Crosses and boxes represent novel and non-novel notes, respectively. This was the history with the median ROC AUC score for the `NM-NACD`, meaning that it is neither one of the best nor one of the worst showcases of `NM-NACD`. This patient had many simple problems such as cuts, sprains and swollen eyes, most of which seemed to be resolved very quickly. He or she also had a recurring knee problem that evolved through several diagnosis codes, and a back problem giving rise to the series of non-novel notes ending at contact 30.

3.3 The Measure Candidates

In this section, we describe four groups of novelty measures: first four compression-based measures, then the baseline measure from the related field of new event detection, then an SVM with all the other measures as input and then six naive measures to serve as a reality check for the others.

Measures Based on Text Compression. The NCD, as defined above, can be used in several ways to construct a novelty measure. One way, `NM-NCD` is to use the NCD between the concatenation of all the previous notes and the current note:

$$\texttt{NM-NCD}(p,k) = \mathrm{NCD}(\underset{i\in[1,k)}{\mathrm{concat}}(n_i^p), n_k^p) \tag{4}$$

This formulation has the problem that the distances typically will increase as more notes are added to the concatenation.

Another way is to take the minimum of the NCDs between the current note and the previous notes, reasoning that the current note is novel if it is not similar to any of the previous notes.

$$\texttt{NM-MIN-NCD}(p,k) = \min_{i\in[1,k)} \text{NCD}(n_k^p, n_i^p) \tag{5}$$

The ID, NID and NCD are symmetric distances, since the program must compute x from y and y from x. Since measuring the novelty of a note with respect to a set of other notes is not an inherently symmetric operation, it is relevant to consider asymmetric variants of the distances. The asymmetric counterparts to $ID(x,y)$ are of course $K(x|y)$ and $K(y|x)$. We can normalize $K(y|x)$ by dividing by $K(y)$, obtaining an asymmetric distance that is 0 when x contains all the information about y and 1 when x contains none of the information about y. We call this distance the normalized, asymmetric information distance, the NAID:

$$\text{NAID}(x,y) = \frac{K(y|x)}{K(y)} \tag{6}$$

Note that the NAID, as opposed to the NID, doesn't take into account the amount of information x contains unrelated to y. This might be beneficial if we think that unrelated information per definition is irrelevant for the question of y's novelty. The NAID, unlike the NID, is not a metric, but our application does not require that property.

In analogy with the way [2] approximated the NID by the NCD, we approximate the NAID by the NACD:

$$\text{NACD}(x,y) = \frac{C(xy) - C(x)}{C(y)} \tag{7}$$

We use the NACD as a novelty measure by applying it from the concatenation of the previous notes to the current note:

$$\texttt{NM-NACD}(p,k) = \text{NACD}(\underset{i\in[1,k)}{\text{concat}}(n_i^p), n_k^p) \tag{8}$$

As opposed to the NCD, the NACD will not in general increase as more notes are added, because the information in the past notes that do not contribute to better compression of the n_k^p is not taken into account.

We can also take the minimum of the NACD from each of the previous notes to the current note:

$$\texttt{NM-MIN-NACD}(p,k) = \min_{i\in[1,k)} \text{NACD}(n_k^p, n_i^p) \tag{9}$$

We used `gzip` as the compressor C in our experiments, at the maximum compression level (9).

We have also tested un-normalized versions of the compression-based novelty measures. They always performed worse than their normalized counterparts. For brevity, we omit them from the results.

Vector Space Measures. We transplanted the incremental TF-IDF vector space scheme, considered a baseline in New Event Detection, to our problem. The scheme can be formulated as using a function $\mathrm{tf}(w, n)$, which gives the frequency with which the word w occurs in the document d, and a function $\mathrm{df}(w, N)$, which gives the fraction of the notes in the set of notes N that contain the word w. For a prediction problem (p, k), let $N_k^p = \{n_i^{p'} : p' \neq p \vee i \in [1, k)\}$, (the set of notes available in this and other histories), and let $W_k^p = \{w : \mathrm{df}(w, N_k^p) >= \theta\}$ (the set of words under consideration, excluding very rare words). Following [8], we use $\theta = 2$. Represent a note n with a vector of weights, one for each word w, with the following (the tf-idf terms):

$$\mathrm{weight}_k^p(w, n) = \mathrm{tf}(w, n) \frac{1}{\mathrm{df}(w, N_k^p)}. \tag{10}$$

Define similarity between two notes n_a and n_a with the following (the cosine distance):

$$\mathrm{sim}_k^p(n_a, n_b) = \frac{\sum_{w \in W_k^p} \mathrm{weight}_k^p(w, n_a)\, \mathrm{weight}_k^p(w, n_b)}{\sqrt{\sum_{w \in W_k^p} |\, \mathrm{weight}_k^p(w, n_a)\, \mathrm{weight}_k^p(w, n_b)|^2}} \tag{11}$$

Then, take the following as a novelty measure:

$$\texttt{NM-TFIDF}(p, k) = 1 - \max_{i \in [1,k)} \{\mathrm{sim}(n_k^p, n_i^p)\} \tag{12}$$

In calculating df and tf the pre-processing went as follows: First, all non-alphanumeric characters were replaced by white-space. Then, the words were found by splitting the string on white-space. Spelled-out numbers were replaced by numbers, stop words were removed, and naive, dictionary-based stemming was applied.

Combined Measure. We combined the eleven other measures mentioned above as features to a support vector machine (SVM), and used the output of the SVM as the novelty measure `NM-SVM`. We used libsvm's [11] implementation of C-class SVM with default parameters (3rd degree radial basis kernel, $C = 1$, $\gamma = 1/11$). The negative examples were sampled down to the number of positive examples.

Trivial Measures. We also implemented six trivial novelty measures, in order to check whether the other measures were doing something more than this. The measures were `NM-RANK` (the position of the note relative to the beginning of the history), `NM-DAY`(the day of the note relative the the beginning of the history, `NM-DAYGAP` (the number of days since the previous note), `NM-#CHARS` (the number of characters in the note), `NM-#NW` (the number of previously unseen words in the note, and `NM-C(n)` (the compressed size of the note).

3.4 Experimental Setup

As formalized in subsection 3.2, we extracted patient histories from the data source and calculated the value of each novelty measure for each note in each

history. For the SVM, we did 5-fold cross-validation and used the SVM output on the testing folds as novelty predictions. To check the contribution of the features, we also did 13 other SVM runs, one with all features and 12 each with a different feature left out. Here we only did 2-fold cross-validation.

By setting a threshold for a measure's novelty values, one can obtain a novelty detector with a desired sensitivity/specificity trade-off. The possible trade-offs can be visualized with receiver operating characteristic (ROC) curves. We used the R package ROCR [12] to calculate the ROC) for our measures, and the areas under the curves (AUC). We use the AUCs to score the goodness of the measures. We used DeLong's placement value method [13,14] to calculate p-values for the differences between the AUCs.

4 Results

Figure 2 shows the ROC curves for the measures. Tables 2 and 3 show the AUC scores for the non-trivial and the trivial measures, respectively. `NM-SVM`'s AUC is larger than each of the other AUCs with p-values < 0.0001. `NM-NACD`'s AUC is larger than each of the other AUCs except `NM-SVM`'s with p-values < 0.0001. Leaving out single features did not improve performance of the SVM, beyond perturbations in the fourth decimal of the AUC.

Table 2. AUCs for the non-trivial novelty measures. * means that the measure was inverted, since it gave AUC < 0.5.

NM-NCD*	NM-MIN-NCD	NM-NACD	NM-MIN-NACD	NM-TFIDF	NM-SVM
0.600	0.780	0.826	0.798	0.795	0.850

Table 3. AUCs for the trivial novelty measures. * means that the measure was inverted, since it gave AUC < 0.5.

NM-RANK*	NM-DAY	NM-DAYGAP	NM-#CHARS	NM-#NW	NM-C(n)
0.575	0.576	0.656	0.0.727	0.722	0.735

5 Discussion

We will now attempt to make sense of the results. It is no surprise that the SVM had the best performance since it could combine the information represented in all the other measures. The fact that the asymmetric compression-based measures perform better than their symmetric counterparts seems to confirm the idea that motivated them in the methods section, namely that it doesn't matter for the novelty of the new note how much or little unrelated information is present in the past notes. What matters is the proportion of new information in the new note itself. `NM-NCD`'s worse-than-random performance is probably due to the fact that the complexity of the concatenated previous notes increases as the history

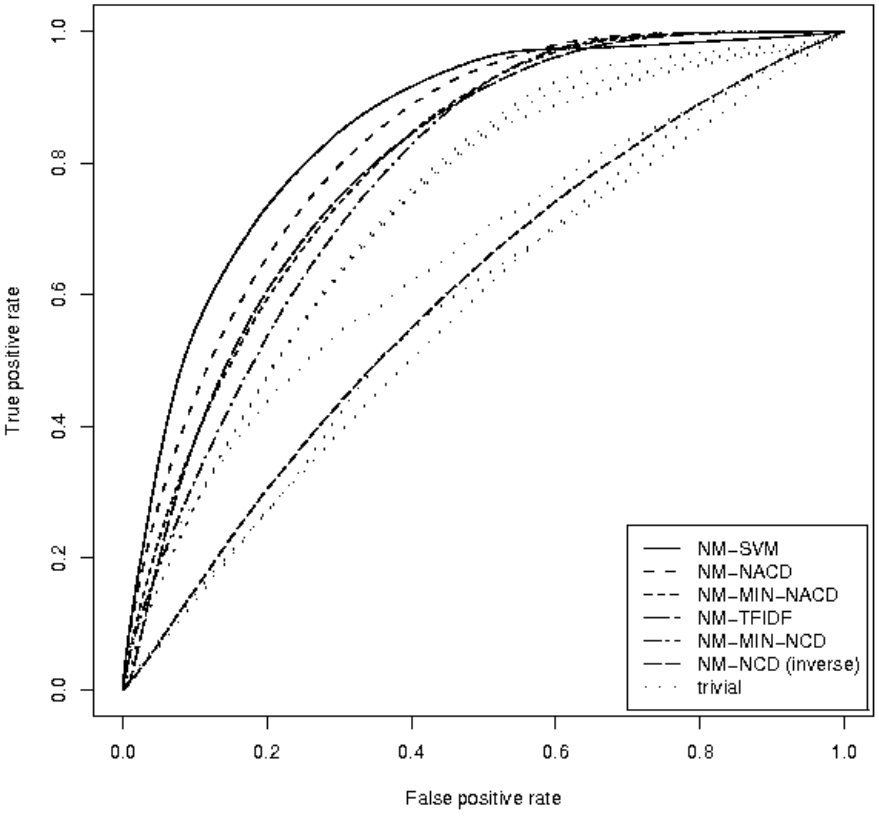


Fig. 2. ROC curves for the novelty measures. (The curves for the trivial measures are all shown with the same line type).

progresses, while the frequency of new diagnosis codes decreases (since more and more codes have already been used). Why `NM-NACD` outperformed `NM-MIN-NACD` is less certain. Perhaps the minimum function made it more susceptible to noise. Perhaps compressing the new note together with all the previous notes at once eliminates more of the non-novel content in the new note, leaving a more accurate estimate of the novelty. Except for the `NM-NCD`, all the non-trivial measures performed better than all the trivial ones, indicating that they did more than just pick up on trivial features.

In comparing the performance of `NM-NACD` to the transplanted, TF-IDF-based method, we must remember that our implementation of `NM-TFIDF` was very primitive. If better natural language processing tools, such as taggers for word categories, were available, `NM-TFIDF` might do better. Following the example of [9], we could use separate TF-IDF vectors for different categories of clinical terms. Or, following [7], we could first classify the notes into broad categories, and then specialize the document representation for each category.

Another question is how good the best measure, `NM-SVM`, is. Looking at the ROC curves in figure 2, it is clear that we cannot find a trade-off that allows us to detect most of the novel notes without incurring a large rate of false alarms. Performance must be improved before a clinically useful tool can be created. One avenue of possible improvement is to move beyond the textual notes and take other data types into account. For example, a novelty detector based on the textual notes *and* the diagnosis codes could classify as novel all encounters with a new diagnosis code and/or text-based novelty above a given threshold.

We have reached a point where a new evaluation criterion is needed. The diagnosis-based criterion gave us a massive data set for comparing text-based measures, but if the novelty detector incorporates the diagnosis codes, the criterion becomes invalid. Furthermore, the diagnosis-based criterion is only an imperfect reflection of our ultimate criterion: novelty in the eyes of clinicians. A natural next step is therefore to have clinicians manually rate the novelty of each note in a sufficiently large set of patient histories. With the long and complex histories typical of general practice, this will not be a small undertaking.

6 Conclusion

We have defined and evaluated several text-based measures for detecting novel encounter notes in patient histories. The best single measure was `NM-NACD`, the asymmetric normalized compression distance, proposed by us in this article. It was only slightly outperformed by an SVM using all single measures as features. This lets us conclude that compression-based measures have utility for detecting novelty in text, and that asymmetry, while disadvantageous in other situations, may be an advantage for the asymmetric comparisons needed for novelty detection. The text-based measures did not perform well enough to be clinically useful on their own, but would likely by useful as components in a system detecting novelty from a wider range of data types.

Acknowledgements

Part of this work was funded by the Norwegian Research Council (NFR). The authors would like to thank Anders Grimsmo and Carl-Fredrik Bassøe for valuable assistance, discussions and comments.

References

1. Bennett, C.H., Gács, P., Li, M., Vitányi, P.M.B., Zurek, W.H.: Information distance. IEEE Transactions on Information Theory 44(4), 1407–1423 (1998)
2. Li, M., Chen, X., Li, X., Ma, B., Vitányi, P.M.B.: The similarity metric. IEEE Transactions on information theory 50(12), 3250–3264 (2004)
3. Keogh, E., Lonardi, S., Ratanamahatana, C.A.: Towards parameter-free data mining. In: KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 206–215. ACM Press, New York, NY, USA (2004)

4. Cilibrasi, R., Vitányi, P.M.B.: Clustering by compression. IEEE Transactions of Information Theory 51(4), 1523–1545 (2005)
5. Grumbach, S., Tahi, F.: A new challenge for compression algorithms. Information Processing and Management 30, 875–866 (1994)
6. Varré, J.-S., Delahaye, J.-P., Rivals, E.: Transformation distances: a familty of dissimilarity measures based on movements of segments. Bioinformatics 15(3), 194–202 (1999)
7. Yang, Y., Zhang, J., Carbonell, J., Jin, C.: Topic-conditioned novelty detection. In: KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, pp. 688–693. ACM Press, New York (2002)
8. Brants, T., Chen, F.: A system for new event detection. In: SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, New York, NY, USA, pp. 330–337. ACM Press, New York (2003)
9. Kumaran, G., Allan, J.: Using names and topics for new event detection. In: HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Morristown, NJ, USA, pp. 121–128. Association for Computational Linguistics (2005)
10. WONCA. ICPC-2: International Classification of Primary Care. WONCA International Classification Committee, Oxford University Press (1998)
11. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001), Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
12. Sing, T., Sander, O., Beerenwinkel, N., Lengauer, T.: Rocr: visualizing classifier performance in R. Bioinformatics 21(20), 3940–3941 (2005)
13. DeLong, E.R., DeLong, D.M., Clarke-Pearson, D.L.: Comparing the areas under two or more correlated receiver operating characteristic curves: A nonparametric approach. Biometrics 44, 837–845 (1988)
14. Hanley, J.A., Hajian-Tilaki, K.O.: Sampling variability of nonparametric estimates of the areas under receiver operating characteristic curves: An update. Statistics in Radiology 4, 49–58 (1997)

Author Index