Minho Lee Akira Hirose Zeng-Guang Hou
Rhee Man Kil (Eds.)

LNCS 8227

Neural Information Processing

20th International Conference, ICONIP 2013
Daegu, Korea, November 2013
Proceedings, Part II

2 Part II

Springer

Lecture Notes in Computer Science 8227

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

Minho Lee Akira Hirose Zeng-Guang Hou
Rhee Man Kil (Eds.)

Neural Information Processing

20th International Conference, ICONIP 2013
Daegu, Korea, November 3-7, 2013
Proceedings, Part II

Springer

Volume Editors

Minho Lee
Kyungpook National University
1370 Sankyuk-Dong, Puk-Gu, Taegu 702-701, Korea
E-mail: mholee@knu.ac.kr

Akira Hirose
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
E-mail: ahirose@ee.t.u-tokyo.ac.jp

Zeng-Guang Hou
Chinese Academy of Sciences, Institute of Automation
Key Laboratory of Complex Systems and Intelligence Science
Beijing 100190, China
E-mail: zengguang.hou@ia.ac.cn

Rhee Man Kil
Sungkyunkwan University
2066, Seobu-ro, Jangan-gu, Suwon 440-746, Korea
E-mail: rmkil@skku.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-42041-2 e-ISBN 978-3-642-42042-9
DOI 10.1007/978-3-642-42042-9
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013951838

CR Subject Classification (1998): I.5, I.4, I.2, F.1, H.3, H.4, F.2, G.3, J.3, C.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume is part of the three-volume proceedings of the 20th International Conference on Neural Information Processing (ICONIP 2013), which was held in Daegu, Korea, during November 3–7, 2013. ICONIP is the annual conference of the Asia Pacific Neural Network Assembly (APNNA). This series of conferences has been held annually since ICONIP 1994 in Seoul and has become one of the premier international conferences in the areas of neural networks.

Over the past few decades, the neural information processing community has witnessed tremendous efforts and developments from all aspects of neural information processing research. These include theoretical foundations, architectures and network organizations, modeling and simulation, empirical study, as well as a wide range of applications across different domains. Recent developments in science and technology, including neuroscience, computer science, cognitive science, nano-technologies, and engineering design, among others, have provided significant new understandings and technological solutions to move neural information processing research toward the development of complex, large-scale, and networked brain-like intelligent systems. This long-term goal can only be achieved with continuous efforts from the community to seriously investigate different issues of the neural information processing and related fields. To this end, ICONIP 2013 provided a powerful platform for the community to share their latest research results, to discuss critical future research directions, to stimulate innovative research ideas, as well as to facilitate multidisciplinary collaborations worldwide.

ICONIP 2013 received tremendous submissions authored by scholars coming from 30 countries and regions across six continents. Based on a rigorous peer review process, where each submission was evaluated by at least two qualified reviewers, about 270 high-quality papers were selected for publication in the prestigious series of *Lecture Notes in Computer Science*. These papers cover all major topics of theoretical research, empirical study, and applications of neural information processing research.

In addition to the contributed papers, the ICONIP 2013 technical program included a keynote speech by Shun-Ichi Amari (RIKEN Brain Science Institute, Japan), 5 plenary speeches by Yoshua Bengio (University of Montreal, Canada), Kunihiko Fukushima (Fuzzy Logic Systems Institute, Fukuoka, Japan), Soo-Young Lee (Brain Science Research Center, KAIST, Korea), Naftali Tishby (The Hebrew University, Jerusalem, Israel) and Zongben Xu (Xi'an Jiatong University, China). This conference also featured invited presentations, regular sessions with oral and poster presentations, and special sessions and tutorials on topics of current interest.

Our conference would not have been successful without the generous patronage of our sponsors. We are most grateful to our sponsors Korean Brain Research

Institute, Qualcomm Korea. We would also like to express our sincere thanks to the International Neural Network Society, European Neural Network Society, Japanese Neural Network Society, Brain Engineering Society of Korea, and The Korean Society for Cognitive Science for technical sponsorship.

We would also like to sincerely thank honorary chair Shun-ichi Amari, Soo-Young Lee, the members of the Advisory Committee, the APNNA Governing Board and past presidents for their guidance, the organizing chair Hyeyoung Park, the members of the Organizing Committee, special sessions chairs, Publication Committee and publicity chairs, for all their great efforts and time in organizing such an event. We would also like to take this opportunity to express our deepest gratitude to the members of the Program Committee and all reviewers for their professional review of the papers. Their expertise guaranteed the high quality of the technical program of the ICONIP 2013!

Furthermore, we would also like to thank Springer for publishing the proceedings in the prestigious series of *Lecture Notes in Computer Science*. We would, moreover, like to express our heartfelt appreciation to the keynote, plenary, panel, and invited speakers for their vision and discussions on the latest.

Finally, we would like to thank all the speakers, authors, and participants for their great contribution and support that made ICONIP 2013 a huge success.

This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government.

November 2013

Minho Lee
Akira Hirose
Rhee Man Kil
Zeng-Guang Hou

Organization

Honorary Chair

Shun-ichi Amari	RIKEN, Japan
Soo-Young Lee	KAIST, Korea

General Chair

Minho Lee	Kyungpook National University, Korea

Program Chair

Akira Hirose	The University of Tokyo, Japan
Zeng-Guang Hou	The Chinese Academy of Sciences, China
Rhee Man Kil	Sungkyunkwan University, Korea

Organizing Chair

Hyeyoung Park	Kyungpook National University, Korea

Workshop Chair

Daijin Kim	POSTECH, Korea
Kyunghwan Kim	NT Research, Korea
Seong-Whan Lee	Korea University, Korea

Special Session Chair

Sung-Bae Cho	Yonsei University, Korea
Seiichi Ozawa	Kobe University, Japan
Liqing Zhang	Shanghai Jiao Tong University, China

Tutorial Chair

Seungjin Choi	POSTECH, Korea

Publication Chair

Yoonsuck Choe	Texas A&M University, USA
Hyung-Min Park	Sogang University, Korea
Seong-Bae Park	Kyungpook National University, Korea

Publicity Chair

Kazushi Ikeda	NAIST, Japan
Chi-Sing Leung	University of Hong Kong, Hong Kong
Shaoning Pang	Unitec Institute of Technology, New Zealand

Registration Chair

Min-Young Kim	Kyungpook National University, Korea

Financial Chair

Sang-Woo Ban	Dongguk University, Korea

Local Arrangement Chair

Doo-Hyun Choi	Kyungpook National University, Korea
Jong-Seok Lee	Yonsei University, Korea
Rammohan Mallipeddi	Kyungpook National University, Korea

Advisory Committee

Jonathan H. Chan, Thailand
Wlodzislaw Duch, Poland
Kunihiko Fukushima, Japan
Tom Gedeon, Australia
Aike Guo, China
Akira Iwata, Japan
Nik Kasabov, New Zealand
Irwin King, Hong Kong
Noboru Onishi, Japan
Ron Son, USA
Il Hong Suh, Korea
Shiro Usui, Japan
DeLiang Wang, USA
Lipo Wang, Singapore
Jun Wang, Hong Kong
Lei Xu, Hong Kong
Takeshi Yamakawa, Japan
Byoung-Tak Zhang, Korea
Li-Ming Zhang, China

Program Committee Members

Tani Jun
Soo-Young Lee
Sung-Bae Cho
Sungmoon jeong
Kyung-Joong Kim
C.K. Loo
Nung Kion Lee
Shan He
Dae-Shik Kim
Rubin Wang
Xin Yao
S. Ma
Honghai Liu
Joarder Kamruzzaman
Mallipeddi Rammohan
Zhirong Yang
Anto Satriyo Nugroho
Nikola Kasabov

Jonghwan Lee
Yaochu Jin
DaeEun Kim
Tingwen Huang
Fangxiang Wu
Dongbing Gu
Hongli Dong
Cesare Alippi
Kyung Hwan Kim
Lae-Jeong Park
Sang-Woong Lee
Sabri Arik
Chee-Peng Lim
Haibo He
Dat Tran
Kee-Eung Kim
Seungjin Choi
Robert (Bob) McKay
Xueyi (Frank) Wang
Jennie Si
Markus Koskela
Ivo Bukovský
Ryo Saegusa
El-Sayed El-Alfy
Hyeyoung Park
Bunthit Watanapa
Vinh Nguyen
Kalyana C. Veluvolu
Mufti Mahmud
Gil-Jin Jang
Hyung-Min Park
Jeounghoon Kim
Rhee Man Kil
Sang-Woo Ban
Masa Takatsuka
Chee Seng Chan
Pau-Choo Chung
Uvais Qidwai
Dong-Joo Kim
JongJin Lim
Sungoh Kwon
Long Cheng
Akitoshi Hanazawa
Andrzej Cichocki
Sheng Li
Oclay Kursun
Michel Verleysen
Peter Erdi
Qingsong Song
Bin Li
Huaguang Zhang
Derong Liu
Eric Matson
Mehdi Roopaei
Jacek Ma'ndziuk
Yang Shi
Zhiwu Lu
Xiaofeng Liao
Zhigang Zeng
Ding-Xuan Zhou
James Tin-Yau Kwok
Hsuan-Tien Lin
Osman Elgawi
Chao Zhang
Bo Shen
Nistor Grozavu
Younès Bennani
Jinde Cao
Li-Po Wang
Justin Dauwels
Andrew Leung
Bao-Liang Lu
Changyin Sun
Hong Yan
Abdesselam Bouzerdoum
Emili Balaguer Ballester
H. Tang
Roland Goecke
Jose Alfredo Ferreira Costa
Shri Rai
Kuntal Ghosh
TaeHyun Hwang
Alexander Rast
Yangming Li
Akira Hirose
Akira Iwata
Ko Sakai
Koichiro Yamauchi

Atsushi Shimada
Eiji Uchino
Hayao Shouno
Hayaru Shouno
Heizo Tokutaka
Hideki Asoh
Hiroaki Gomi
Hiroshi Dozono
Hiroshi Kage
Hiroshi Yamakawa
Hiroyuki Nakahara
Ikuko Nishikawa
Jinglu Hu
Jun Nishii
Katsumi Tateno
Katsunari Shibata
Kazuho Watanabe
Kazushi Ikeda
Kazuyuki Samejima
Keisuke Kameyama
Kenichi Tanaka
Kenichiro Miura
Kenji Doya
Kiyohisa Natsume
Masafumi Hagiwara
Michio Niwano
Mingcong Deng
Naoyuki Sato
Noboru Ohnishi
Seiichi Ozawa
Shin Ishii
Shinichiro Kano
Shunji Satoh
Shunshoku Kanae
Takashi Morie
Takashi Omori
Takeshi Aihara
Takio Kurita
Tao Ban
Tetsuya Yagi
Tohru Nitta
Tom Shibata
Toshiaki Omori
Toshihisa Tanaka
Yen-Wei Chen
Yoko Yamaguchi
Yoshikazu Washizawa

Table of Contents – Part II

An Improved ISOMAP for Visualization and Classification of Multiple Manifolds

Wang Hong-Yuan[1,*,**], Ding Xiu-Jie[1], Cheng Qi-Cai[1], and Chen Fu-Hua[2]

[1] School of Information Science & Technology,
Changzhou University, Changzhou 213164, China
hywang@cczu.edu.cn, {showjie1989,caiqi1123}@163.com
[2] Department of Natural Science and Mathematics, West Liberty University,
West Virginia 26074, USA
fuhua.chen@westliberty.edu

Abstract. The classical algorithm ISOMAP can find the intrinsic low-dimensional structures hidden in high-dimensional data uniformly distributed on or around a single manifold. But if the data are sampled from multi-class, each of which corresponds to an independent manifold, and clusters formed by data points belonging to each class are separated away, several disconnected neighborhood graphs will occur, which leads to the failure of ISOMAP. Moreover, ISOMAP behaves in an unsupervised manner and therefore works less effectively for classification. In this paper, two improved versions of ISOMAP, namely Multi-Class Multi-Manifold ISOMAP (MCMM-ISOMAP) for data visualization and ISOMAP for Classification (ISOMAP-C), are proposed respectively. MCMM-ISOMAP constructs a single neighborhood graph, named a between-class neighborhood graph by connection of between-class points with shortest distance of each within-class neighborhood graph, and then ISOMAP algorithm is applied to find the intrinsic low-dimensional embedding structure. ISOMAP-C is essentially an extension of MCMM-ISOMAP to a supervised manner, which is multiplied by scaling factor greater than one so that low dimensional data set after mapping become more compact within class and more separate between classes. Finally, the mapping function from original high dimensional space to low dimensional space can be approximately modeled using Back-Propagation neural network combined with genetic algorithm. Experimental results using MCMM-ISOMAP on synthetic and real data reveal its effectiveness and ones using ISOMAP-C show that the performance is greatly enhanced and robust to noisy data.

Keywords: multiple manifolds, classification, data visualization, ISOMAP, genetic algorithm.

1 Introduction

In the appearance-based face recognition, a 64×64 face image is represented as a real vector in 4096-dimensional Euclidean space. However, in practical applications, the

* Corresponding author.
** Ph.D, Professor, Interesting Area: Pattern Recognition and Intelligence system.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 1–12, 2013.

number of face images available possibly ranges from hundreds to thousands, which is too sparse in so high-dimensional space, “the curse-of-dimensionality” problem exists(Donoho, 2000). Dimensionality reduction techniques are one of the effective solutions to this problem. In general, dimensionality reduction techniques are divided into linear and nonlinear dimensionality reduction. Classical linear dimensionality reduction algorithms include Principal Component Analysis (PCA) (Turk and Pentland, 1991), Independent Component Analysis (ICA) (Comon, 1994), Multi-Dimensional Scaling (MDS) (Cox and Cox, 1994), Linear Discriminant Analysis (LDA) (Duda et al., 2001) and so on. The limitations of linear dimensionality reduction are based on the assumption that the data structure is globally linear, but in practice, a lot of data are more complex and have nonlinear structure. To deal with these data of nonlinear structure, nonlinear dimensionality reduction algorithms have been proposed, such as Self-Organizing Mapping (SOM) (Kohonen, 2001), Principal Curves (Kegl et al., 2000), Generative Topographic Mapping (GTM) (Bishop et al., 1998), Kernel PCA (Scholkopf et al., 1998) and so on. However, these methods often suffer from the difficulties in designing cost functions or tuning too many free parameters. Moreover, most of these methods are computationally expensive, and do not explicitly consider the structure of low-dimensional manifold on or around which the data possibly lie, thus limiting their utility in high-dimensional data sets (Lin and Zha, 2008). In recent years, nonlinear dimensionality reduction, based on the assumption that the data lie on or around a single low-dimensional manifold in a high-dimensional Euclidean space, has become very popular in machine learning compute r vision and other areas, and many manifold learning algorithms have been proposed, such as ISOmetric feature MAPping (ISOMAP) (Tenenbaum et al., 2000), Locally Linear Embedding (LLE) (Roweis and Saul, 2000), Laplacian Eigenmaps (LE) (Belkin and Niyogi, 2003), Locality Preserving Projection (LLP) (He and Niyogi., 2004) , and Local Tangent Space Alignment (LTSA) (Zhang and Zha, 2005). However, such manifold learning algorithms succeed only if the data lie on or around a single manifold. In practical applications, the sampled data may be very complicated, which comes from more than one manifold whose intrinsic dimension is the same or different. At this point the above-mentioned manifold learning algorithms fail to achieve the correct low-dimensional embedding. Though the related research of multiple manifolds has been reported abroad in the past several years (Goldberg and Zhu; Wang et al., 2008), the proposed algorithm named Multi-Class Multi-Manifold ISOMAP (MCMM-ISOMAP) is an improvement of the classical ISOMAP algorithm, whose contribution is visualization of multiple separate manifolds, each for one class, in a single global coordinate system.Moreover, ISOMAP works in an unsupervised manner and therefore it is unsuitable for classification task. Therefore, the other contribution of this paper is to propose ISOMAP for Classification (ISOMAP-C), which is an extension of MCMM-ISOMAP from an unsupervised manner to a supervised one for the purpose of classification.

The rest of the paper is organized as follows. In section 2, the principle of ISOMAP and its shortcomings are briefly introduced. In section 3, MCMM-ISOMAP is proposed, and the relationship between it and classical ISOMAP are discussed. In section 4, ISOMAP-C is proposed. The experimental results are given in section 5. Finally in section 6, conclusions are drawn and several issues for the future work are indicated.

2 Overview of ISOMAP

In order to measure the true distance between the points lying on a single nonlinear manifold, ISOMAP adopts the shortest path between the data points in a weighted undirected graph formed by data points (Bernstein et al., 2000). Characteristics (disadvantages) of ISOMAP are as follows:

(a) When data is sampled from multiple faraway clusters, each corresponding to a manifold, it is very likely that the short circuit edges will occur, which leads the neighborhood graph to not reflect the intrinsic geometry structure of data and therefore yields undesirable embeddings.

(b) When constructing the neighborhood graph, if more than one sub-neighborhood graphs generate, ISOMAP only finds lower dimensional embedding for one of these sub-neighborhood graphs.

(c) ISOMAP works in an unsupervised manner and therefore it is unsuitable for classification task.

(d) ISOMAP can explicitly provide the mapping function from high-dimensional space to low-dimensional feature space mapping and therefore it is difficult to obtain the low-dimensional embedding for an unseen data in original high-dimensional space.

3 MCMM-ISOMAP

For data visualization, the goal is to map data points in original space into a two or three dimensional one while preserving, as much as possible, the intrinsic structure. As described in the above section, when the data are sampled from multi-class, each of which corresponds to an independent underlying manifold, and the clusters formed by data points from each class are separated away, ISOMAP algorithm possibly fails to find the low-dimensional embedding of data in high-dimensional space. However, the proposed algorithm MCMM-ISOMAP can effectively solve the problem.

The main idea of this algorithm is to obtain a single neighborhood graph over all data points not by means of making the value of the neighborhood parameter large, but by selecting an appropriate value of the neighborhood parameter with which each within-manifold will not generate short-circuit edges. Although multiple neighborhood graphs will form over all data points, they will become a single neighborhood graph by finding pairwise data, each of which are two endpoints of the shortest Euclidean distance between classes, and making the two endpoints neighborhood ones. Finally the classical ISOMAP algorithm is used to get the low-dimensional embedding of all data. Consider a set of samples with labels $X = \{(x_1, l_1), \cdots, (x_N, l_N)\}$, $x_i \in R^n$ and $l_i \in \{1, 2, \cdots, c\}$, i.e., there are c classes in total in X. For convenience, three definitions are introduced as follows:

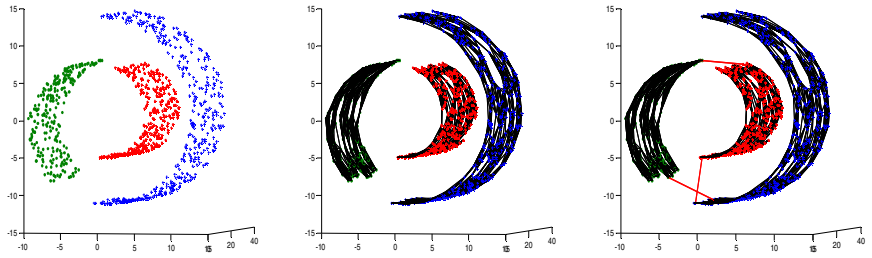
Definition 1: Given a data set with label, a neighborhood graph NG_i is called as the i-th within-class neighborhood graph, if NG_i is constructed on a sub-data set belonging to the i-th class by k nearest neighbors or some fixed radius ε.

According to the above definition, obviously it can be shown that c neighborhood graphs can be constructed on X in total.

Definition 2: For the shortest distance d_{ij} among Euclidean distances between the i-th class and j-th class, the corresponding two endpoints are called as between-class points with shortest distance and the corresponding edge e_{ij} as a between-class adjacent edge.

Definition 3: A neighborhood graph G is said to be a between-class neighborhood graph if G is constructed by connection of c within-class neighborhood graphs with between-class adjacent edges with shortest distance.

The above three definitions are easy to understand intuitively from Figure 1, where Fig.1 (a) shows the cropped Swiss roll data set with two gaps which are categorized as three classes, each denoted by different colors (red, blue and green), three within-class neighborhood graphs are shown in Fig.1 (b) with neighborhood parameter k set to 8, and the between-class neighborhood graph on the data set is shown in Fig.1 (c), where three lines in red are between-class adjacent edges.



(a) (b) (c)

Fig. 1. Swiss roll data sets and neighborhood graph: (a) Swiss roll samples with three classes (red, blue, green). (b) within-class neighborhood graph. (c) between-class neighborhood graph with three between-class adjacent edges (red).

According to class labels, X is split into c disjoint sub-data sets: $X = X_1 \cup X_2 \cup \cdots \cup X_c$, where $X_i = \{x_i^1, x_i^2, \cdots x_i^{N_i}\}$ is the data set of the i-th class and x_i^p $(1 \le p \le N_i)$ indicates the p-th sample belonging to the i-th class. Obviously, $\sum_{i=1}^{c} N_i = N$. Let $NE(x_i^p)$ be a set whose elements are composed of neighbors of x_i^p. Let $Y = \{Y_1, Y_2, \cdots, Y_c\}$ be the lower-dimensional embedding of the input data set. $y_i^p \in Y_i \subset R^m$ indicates the m dimensional embedding of x_i^p,

generally $m << n$. The detailed description of the MCMM-ISOMAP algorithm is as follows:

Step 1. Construct c within-class neighborhood graphs with the appropriate number of neighbors k with which short-circuits can not occur within class.

Step 2. Search for between-class points with shortest distance between classes, which are obtained by solving the equation (1)

$$WM_i, WM_j = \arg \min_{\substack{w_i \in X_i \\ w_j \in X_j}} \min_{\substack{1 \le i, j \le c \\ i < j}} (\left\| X_i - X_j \right\|) \tag{1}$$

where $WM_i \in X_i, WM_j \in X_j$.

i.e. (WM_i, WM_j) are between-class points with shortest distance between the i-th class and the j-th class.

Step 3. Substitute $NE(WM_i)$ and $NE(WM_j)$ with $NE(WM_i) \cup \{WM_j\}$ and $NE(WM_j) \cup \{WM_i\}$, respectively. In this way, a between-class neighborhood graph will form over the entire data set.

Step 4. Compute the shortest path distance between all pairs of data: the shortest path distance between one pair of data usually is computed by the Floyd or Dijkstra algorithm.

Step 5. Construct low-dimensional embedding in m dimensional Euclidean space: the shortest path distance between pairwise data derived from the Step 4 is used as the input of MDS, consequently getting the m dimensional embedding $Y = \{Y_1, Y_2, \cdots, Y_c\}$.

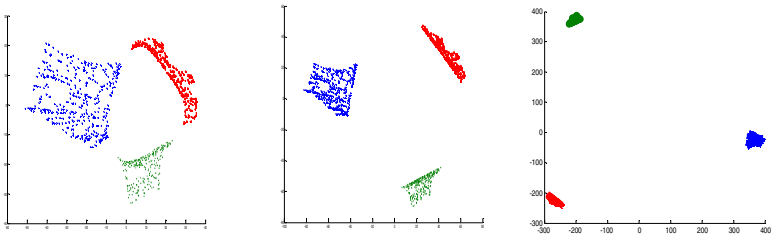
From the above algorithm steps, it can be concluded that the difference and relationship between MCMM-ISOMAP and ISOMAP is as follows:

① MCMM-ISOMAP is the same as ISOMAP in the sense that there is only one parameter to be adjusted in the two algorithms, which indicates the MCMM-ISOMAP algorithm is simply implemented and can be spread easily in practical applications.

② Assuming that the input data sets are sampled from multi-class and that clusters formed by these data points belonging to different classes are closer, if the selected value of the neighborhood parameter can not only guarantee that the short-circuit edges can not be introduced within manifold, but also that a single neighborhood graph will form over the entire data set, the above Step 2 to Step 3 can be omitted. Consequently MCMM-ISOMAP is the same as ISOMAP. In this sense, ISOMAP is a particular case of MCMM-ISOMAP, and MCMM-ISOMAP is an extension of ISOMAP. In other words, whether the clusters of classes are closer or not, MCMM-ISOMAP algorithm can obtain the low-dimensional embeddings of the input data sets.

4 ISOMAP-C

From a classification point of view, it is expected that the classifier can make data belonging to different classes separate clearly, while data belonging to the same class are close in the low-dimensional feature space. Therefore, if the length of between-class adjacent edges is rescaled by multiplication with the scaling factor σ greater than one, approximate geodesic distances between classes will enlarge, which will make the lower dimensional embedding of different classes separate clearly. Obviously, the larger the value of σ, the larger the margin between classes, ultimately making data from the same class shrink to a point when σ approaches infinity. The 2-dimensional embeddings of data in Fig. 1 (a) are shown in Fig. 2 with σ set to 1, 10 and 100, respectively.



(a) $\sigma = 1$ (b) $\sigma = 10$ (c) $\sigma = 100$

Fig. 2. The embedding in 2D space with scaling factor set to different values

It is known that the mapping function f from the original high-dimensional space to low dimensional feature one is not explicitly given by ISOMAP. Therefore, for a test data point to be classified, its low-dimensional embedding can be found by repetition of the ISOMAP procedure, which is an obvious waste of time. In this paper, the mapping function f can be approximated by a Back-Propagation neural network, whose initial weights and thresholds are optimized by Genetic Algorithms to avoid local minimum using gradient decent techniques. The BP neural network is trained by the data set in original space as input and the corresponding low-dimensional embedding as output. ISOMAP-C algorithm is summarized as follows:

Step 1~3: The first three steps are the same as ones in MCMM-ISOMAP.

Step 4: Substitute $d_{ij} = \left\| WM_i - WM_j \right\|$ with $d_{ij} = d_{ij} \times \sigma$ ($\sigma > 1$).

Step 5: Calculate the shortest path between all pairs of data to approximate geodesic distances via Floyd's algorithm or Dijkstra's algorithm, and then MDS is applied to obtain the embedding Y.

Step 6: BP neural network is trained with (X, Y), whose initial weights and thresholds are obtained via genetic algorithm.

Step 7: For a given unseen point x_0, x_0 is used as input of BP neural network and y_0 is obtained as output.

Step 8: k-Nearest neighbors algorithm is applied on Y to predict the class label of x_0.

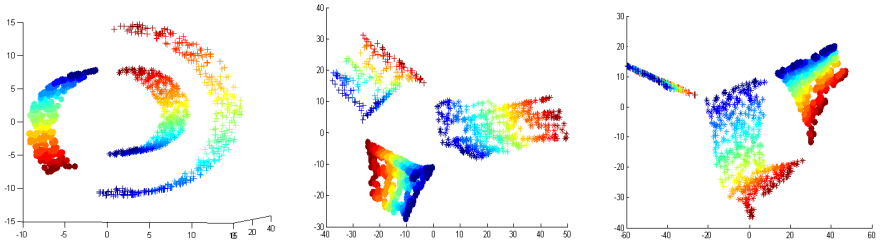
5 Experimental Results

5.1 MCMM-ISOMAP for Data Visualization

In order to test the performance of MCMM-ISOMAP for data visualization, experiments were carried out on synthetic data sets and real-world data sets respectively.

Synthetic Data Sets

The three classes of Swiss roll data, marked by ●,＋ and·respectively, are constructed as shown in Fig.3 (a), where each class corresponds to a two dimensional manifold. When the value of neighborhood parameter k is set to 8, short-circuit edges can not occur in any of the three within-class neighborhood graphs. However, in order to apply ISOMAP to such a data set, the single neighborhood graph will form over the entire input data with the minimum neighborhood size $k=15$. The experimental results using MCMM-ISOMAP and ISOMAP are shown in Fig. 3 (b) and (c), respectively. It can be seen from the results that MCMM-ISOMAP better preserves the topological structure of the data in the same class, but ISOMAP fails since it forces the class, marked by symbol ＋, to shrink to almost a line, and the class, marked by symbol ·, to distort in the middle.



(a) (b) (c)

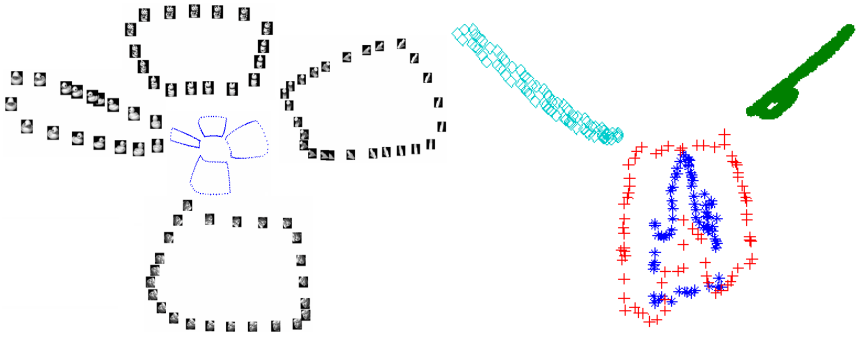
Fig. 3. The result on synthetic data sets: (a) three classes Swiss roll data, (b) using MCMM-ISOMAP, $k=8$, (c) using ISOMAP, $k=15$

Real-World Data Sets

The data sets in this experiment includes four classes of images (duck, building block, cat and face), each of which consists of 72 images which have smooth change in appearance due to pose variations. Data sets of the first three classes duck, building

block and cat can be downloaded from the Computer Vision Laboratory of Columbia University(Nene et al., 1996), and the class face images can be obtained by rotating every 5 degrees a front facial image selected from facial images data set used in ISOMAP[1]. All images were resized to 32*32.

The result of using MCMM-ISOMAP with a neighborhood size $k = 3$ is shown in Fig. 4 (a), where part of typical images of each class are shown around the data points. It can be seen that MCMM-ISOMAP successfully embed images in the four classes into a two-dimensional global coordinate system. However, the minimum neighborhood size is $k = 9$, with which a single neighborhood graph over such four classes of data sets can be constructed. The result using the ISOMAP with $k = 9$ is shown in Fig. 4 (b). It can be seen from the result that ISOMAP fails to find the intrinsic structure of every manifold, because images belonging to the classes duck and face both shrink almost into a straight line, and that both classes of cat and building block drastically intersect after the dimensionality reduction.



(a) (b)

Fig. 4. The result on real-world data sets: (a) MCMM- ISOMAP, $k = 3$ (b) ISOMAP, $k = 9$. (duck, building block, cat and face, marked by symbol ●, +, * and ◇, respectively)

5.2 ISOMAP-C for Data Classification

Data sets used in the experiment are from UCI machine learning repository (Blake and Merz, 1998). All instances with missing values are excluded.. ISOMAP-C method is compared with BP neural network (Rumelhart et al., 1986), RBF neural network (Chen et al., 1991), and K nearest neighbors (Ho, 1998) implemented based on WEKA (Witten and Frank, 2005) with default parameters and SVM (Vapnik, 1995, 1998) implemented based on LIBSVM (Chang and Lin, 2001). The BP neural network to approximate mapping function contains three layers, and the numbers of neurons for input layer, hidden layer and output layer is equal to the dimension n of X, 20 and the dimension m of Y set to 2 indicating that all input data sets are reduced to 2 dimensional feature space. A real coded genetic algorithm is implemented

[1] http://waldron.stanford.edu/~isomap/datasets.html

based on GAOT Toolbox to optimize initial weights and thresholds of the BP neural network. The initial population takes 50.

On each data set, ten-fold cross validation is repeated ten times. The results are shown in Tab. 1. Moreover, in order to evaluate whether the performance between ISOMAP-C and the other classifiers is significantly different, the corrected resample t-test are employed at the significance level 0.05. The results are shown in Tab. 2, where the last row indicates the total of "win/tie/loss" between ISOMAP-C and one algorithm on all datasets and the last column indicates the total of "win/tie/loss" between ISOMAP-C and all the other algorithms on one dataset.

Table 1. The rate of accuracy of each algorithm (in the form of mean (standard deviation)%)

dataset	ISOMAP-C	ISOMAP	BP	RBF	K-NN	C4.5	SVM
bal	**96.49(1.57)**	73.62(4.03)	90.69(3.04)	86.34(3.41)	90.26(1.95)	77.82(3.42)	89.87(1.67)
bre	95.63(1.00)	73.48(3.43)	96.10(2.18)	96.32(2.18)	**96.92(2.08)**	95.44(2.64)	96.33(2.04)
dia	**74.60(2.41)**	69.28(5.17)	74.75(4.90)	74.04(4.91)	72.94(4.26)	74.49(5.27)	65.11(0.34)
gla	**67.40(8.55)**	61.08(9.69)	67.32(8.64)	64.92(9.90)	63.26(8.51)	67.63(9.31)	68.34(8.25)
son	76.81(4.35)	64.35(10.21)	**81.61(8.66)**	72.62(9.91)	75.25(9.91)	73.61(9.34)	64.99(7.66)
swr	**100(0.00)**	99.78(0.17)	**100(0.00)**	99.85(0.62)	**100.00(0.00)**	99.79(0.47)	99.84(0.39)

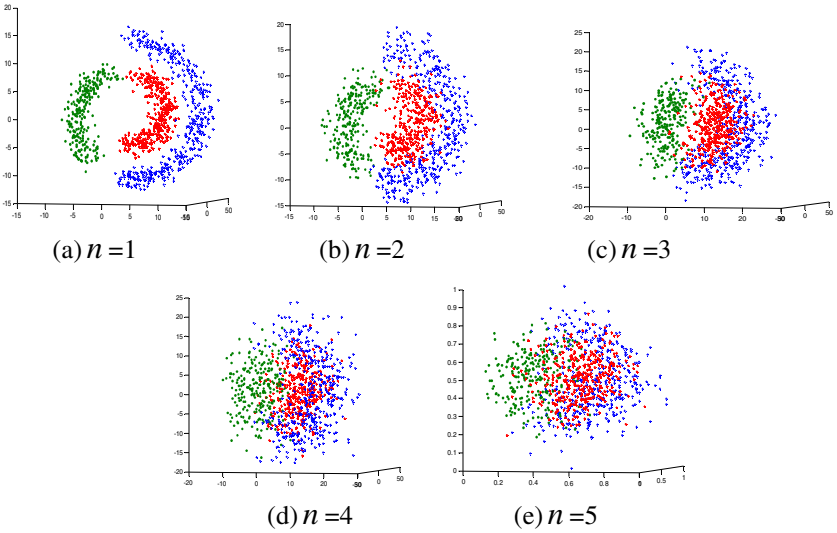
Table 2. The win/tie/loss table comparing ISOMAP-C and other algorithms ("win" means the performance of ISOMAP-C is significantly better, "tie" means the performance is statistically equal, and "loss" means the performance is significantly worse).

dataset	ISOMAP	BP	RBF	K-NN	C4.5	SVM	**win/tie/loss**
bal	*win*	*win*	*win*	*win*	*win*	*win*	**6/0/0**
bre	*win*	*tie*	*tie*	*tie*	*tie*	*tie*	**1/5/0**
dia	*win*	*tie*	*tie*	*tie*	*tie*	*win*	**2/4/0**
gla	*win*	*tie*	*tie*	*tie*	*tie*	*tie*	**1/5/0**
son	*win*	*tie*	*tie*	*tie*	*tie*	*win*	**2/4/0**
swr	*tie*	*tie*	*tie*	*tie*	*tie*	*tie*	**0/6/0**
win/tie/loss	**5/1/0**	**1/5/0**	**1/5/0**	**1/5/0**	**1/5/0**	**3/3/0**	

As seen from Tab. 1, the performance of ISOMAP-C is the best on the four data sets among the above six ones and worse than BP neural network on the sonar data set. It may be due to the number of attributes of the data set, being relatively large, while the number of samples is relatively small, resulting in the excessive loss of information in the procedure of dimension reduction. The last row in Tab. 2 reveals that the performance of ISOMAP-C is not worse than the other algorithms, in particular

significantly better than ISOMAP on five datasets of six ones. And the last column reveals that ISOMAP-C is significantly better than the other algorithms on the balance-scale dataset.

In addition, in order to analyze the robustness of ISOMAP-C for noisy data, we add 1 to 5 times of Gaussian noise with mean 0 and standard deviation 1 to the above Swiss roll data shown in Fig. 5. Results are shown in Tab. 3 using different classifiers on the noisy Swiss roll data via ten times ten-fold cross validation.



(a) $n=1$ (b) $n=2$ (c) $n=3$

(d) $n=4$ (e) $n=5$

Fig. 5. The Swiss roll data plus n times Gaussian noise, with zero mean 0 and standard deviation 1

Table 3. The accuracy of different algorithms on the Swiss roll data set with different times noise (in the form of mean(standard deviation)%)

times	ISOMAP-C	BP	RBF	K-NN	C4.5	SVM
1	99.51(0.53)	99.52(0.68)	98.74(0.79)	**99.71(0.65)**	98.84(1.19)	97.88(2.02)
2	**94.57(1.83)**	93.72(2.86)	92.47(2.08)	93.05(2.02)	92.85(2.64)	86.78(5.19)
3	**84.76(4.43)**	83.49(3.79)	83.01(4.08)	83.78(4.43)	81.95(3.61)	73.65(4.49)
4	75.24(3.79)	**76.13(4.29)**	74.99(4.47)	75.05(4.05)	73.61(3.80)	69.62(4.07)
5	67.30(1.86)	**68.91(5.13)**	67.26(4.44)	66.88(4.49)	67.47(4.94)	51.93(5.21)

It can be shown from Tab. 3 that ISOMAP-C has very good robustness and performance for such a dataset, and the robustness of SVM is the worst. Why ISOMAP-C has good robustness is likely due to the class labels being employed when constructing within-class neighborhood graphs and, therefore, no matter how high the

noise level is, one within-class neighborhood graph is constructed by data only belonging to the same class. In this way, margins between classes are large enough in the lower dimensional space.

6 Conclusions

In this paper, two variants of ISOMAP are proposed: MCMM-ISOMAP for visualization of data lying on or around multiple separate manifolds, each for one class, and ISOMAP-C for data classification. Essentially, ISOMAP-C is an extension of MCMM-ISOMAP from unsupervised manner to supervised one. For an unseen data point to be classified, its low dimensional embedding is found via the approximate mapping function constructed by BP neural network, whose initial weights and thresholds are optimally selected by genetic algorithm. The experimental results using MCMM-ISOMAP on synthetic and real-world data sets indicate it can find the intrinsic topological structure within class, and results using ISOMAP-C indicates that it has good robustness with noisy datasets, surpasses ISOMAP in classification and is highly competitive with those well-known classification methods.

However, disadvantages of such algorithms are that they are time-consuming due to calculation of the shortest path between points, and unsuitable for data sets with categorical attributes and high-dimensional sparse data sets. How to deal with such situations is our future work.

Acknowledgments. This work was partially supported by the National Science Foundation of China under Grant No.61070121 and No.60973094.

References

1. Belkin, M., Niyogi, P.: Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. Neural Computation 15, 1373–1396 (2003)
2. Bernstein, M., de Silva, V., Langford, J.C.: Graph approximations to geodesics on embedded manifolds. Technical Report, Stanford University (2000)
3. Bishop, C.M., Svensen, M., Williams, C.K.I.: GTM: The Generative Topographic Mapping. Neural Computation 10, 215–234 (1998)
4. Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases. University of California. Department of Information and Computer Science 55, Irvine (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001)
6. Chen, S., Cowan, C.F., Grant., P.M.: Orthogonal least squares learning algorithm for radial basis function networks. IEEE Transactions on Neural Networks 2, 302–309 (1991)
7. Comon, P.: Independent component analysis: a new concept? Signal Processing 36, 287–314 (1994)
8. Cox, T., Cox, M.: Multidimensional Scaling. Chapman and Hall (1994)
9. Donoho, D.L.: High-Dimensional Data Analysis: The Curses and Blessings of Dimensionality. In: Proc. AMS Math. Challenges of the 21st Century (2000)
10. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley & Sons (2001)

11. Goldberg, A., Zhu, X.: Multi-Manifold Semi-Supervised Learning. In: Workshop of Multi-Manifold Data Modeling and Applications in IMA
12. He, X., Niyogi., P.: Locality preserving projections. MIT Press (2004)
13. Ho, T.K.: Nearest Neighbors in random subspaces. In: Amin, A., Pudil, P., Dori, D. (eds.) Advances in Pattern Recognition. LNCS, vol. 1451, pp. 640–648. Springer, Heidelberg (1998)
14. Kegl, B., Krzyzak, A., Linder, T., Zeger, K.: Learning and Design of Principal Curves. IEEE Trans. Pattern Anal. Mach. Intell. 22, 281–297 (2000)
15. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer (2001)
16. Lin, T., Zha, H.: Riemannian Manifold Learning. IEEE Trans. Pattern Anal. Mach. Intell. 30, 796–809 (2008)
17. Nene, S., Nayar, S., Murase, H.: Columbia Object Image Library (COIL-20). Columbia University (1996)
18. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290, 2323–2326 (2000)
19. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back propagating errors. Nature 323, 318–362 (1986)
20. Scholkopf, B., Smola, A., Muller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10, 1299–1319 (1998)
21. Tenenbaum, J.B., de Silva, V., Langford., J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290, 2319–2323 (2000)
22. Turk, M., Pentland, A.: Eigenfaces for Recognition. Journal of Cognitive Neuroscience 3, 71–86 (1991)
23. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Berlin (1995)
24. Vapnik, V.: Statistical Learning Theory. John Wiley and Sons, New York (1998)
25. Wang, X., Tiňo, P., Fardal, M.A.: Multiple Manifolds Learning Framework Based on Hierarchical Mixture Density Model. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 566–581. Springer, Heidelberg (2008)
26. Witten, I.H., Frank, E.: Data Mining. Morgan Kaufmann, San Francisco (2005)
27. Zhang, Z., Zha, H.: Principal Manifolds and Nonlinear Dimension Reduction via Local (2005)
28. Tangent Space Alignment. Journal of Scientific Computing 26, 313–338
29. Zhou, Z.H., Cao, C.G.: Neural Networks and Its Applications. Tsinghua University Press (2004)

Probability of Perfect Reconstruction of Pareto Set in Multi-Objective Optimization

Kazushi Ikeda and Akira Hontani

Nara Institute of Science and Technology
Ikoma, Nara 630-0192 Japan
kazushi@is.naist.jp, akira.hontani@gmail.com
http://mi.naist.jp/

Abstract. We proposed a method for collecting all Pareto solutions in multi-objective optimization problems. Our method is similar to randomized algorithms or the statistical learning theory and completely reconstructs the Pareto set with high probability from a finite number of single-objective optimizations. We also derived an upper bound of the mean of the probability that the method fails the perfect reconstruction. Our analysis shows the mean error probability decreases as the number of single-objective optimizations increases.

Keywords: Multi-objective optimization, Pareto set, randomized algorithm, statistical learning theory.

1 Introduction

Multi-objective optimization (MOO) is the class of optimization problems that have plural objective functions [1]. One example is the control of power plants [2] and another example is the design of wing-shape [3]. An MOO problem is formally expressed as

$$\begin{aligned} \max \mathbf{f}(\mathbf{x}) \qquad & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_D(\mathbf{x}))^T, \\ & \text{s.t. } \mathbf{x} \in X \equiv \{\mathbf{x}' | g_k(\mathbf{x}') \leq 0, k = 1, 2, \ldots, K\}, \end{aligned} \tag{1}$$

where $\mathbf{x}$, f_i and g_k denote a vector of design variables, the ith objective function and the jth constraint function. X is called the feasible design space and

$$Z \equiv \{\mathbf{f}(\mathbf{x}) | \mathbf{x} \in X\} \tag{2}$$

is called the feasible criterion space. Note that the maximization of a vector function cannot be defined due to trade-offs.

To formulate (1) mathematically rigorously, the Pareto optimality has been introduced [1]. A point $\mathbf{x}^* \in X$ is Pareto optimal if and only if

$$\nexists \mathbf{x} \in X \text{ s.t. } \mathbf{f}(\mathbf{x}) < \mathbf{f}(\mathbf{x}^*), \exists i, f_i(\mathbf{x}) < f_i(\mathbf{x}^*), \tag{3}$$

where the inequality of a vector takes elementwise. All Pareto optimal points lie on the boundary of Z, which is called the Pareto set (Fig. 1). The existing

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 13–18, 2013.

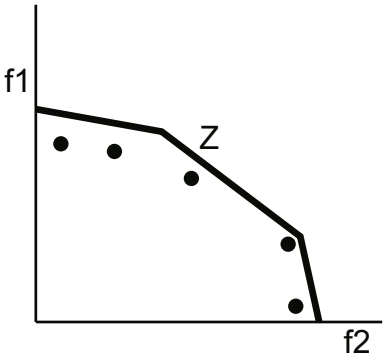


Fig. 1. An illustration. Feasible criterion space (pentagon) and the Pareto set (thick line). Points (black circles) may approximate the Pareto set.

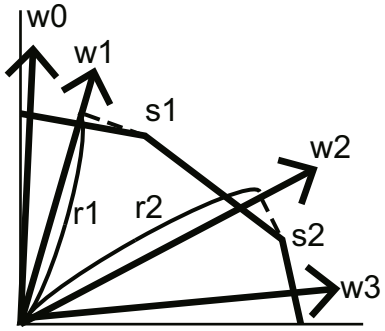


Fig. 2. Optimization of a sum of objective functions with weight $\mathbf{w}$

methods for obtaining the Pareto set approximate the set by particles (points) [4–6]. However, there are few algorithms that have theoretical assurance in approximation accuracy [7]. This difficulty results from the fact that infinite particles are necessary to express the Pareto set exactly.

Consider the weighted sum of given D objective functions for a given weight $\mathbf{w} \in R^D$. Then, one point $\mathbf{s}(\mathbf{w})$ in the Pareto set is found by optimizing this single-objective function according to $\mathbf{w}$ with the distance $r(\mathbf{w})$ (Fig. 2). The function $r : \mathbf{w} \mapsto r(\mathbf{w})$ is a kind of the Legendre transformation when Z is convex. Hence, the Pareto set can be reconstructed if $r(\mathbf{w})$ is known for all $\mathbf{w}$. The Pareto set can be reconstructed from a finite set $\{s(\mathbf{w}_i)\}_{i=1}^N$ if the set is a polyhedron. In Fig. 2, for example, four times of single-objective optimizations with $\mathbf{w}_0, \ldots, \mathbf{w}_3$ find four vertices of Z and hence the Pareto set.

In this study, we proposed a method for reconstructing the Pareto set using a finite times of single-objective optimizations and analyzed its performance. Our method chooses $\{\mathbf{w}_n\}_{n=1}^N$ at random since we have no prior knowledge. The method succeeds to reconstruct in some cases and it does not in others but the

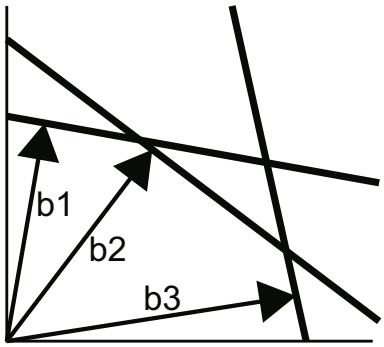


Fig. 3. Constraints are linear functions characterized by $\mathbf{b}_k$

success rate increases N increases. Note that this idea is similar to randomized algorithms [8] or probably-approximately correct (PAC) learning [9, 10].

2 Problem Statement

Suppose D objective functions and K constraint functions are given, where the kth constraint function is a hyperplane,

$$\mathbf{b}_k^T \mathbf{f} = \|\mathbf{b}_k\|^2, \tag{4}$$

that is, the hyperplane with normal vector $\mathbf{b}_k$ (Fig. 3). We assume each $\mathbf{b}_k$ is chosen according to the normal distribution $N(0, I)$ in the first quadrant, that is, its angle θ_k obeys the uniform distribution $U(S_+^{D-1})$ and its length $|\mathbf{b}_k|$ obeys the χ^2 distribution, where S_+^{D-1} is the first quadrant of the $(D-1)$-dimensional sphere. Then, the feasible criterion space Z is a polyhedron that consists of $M+1$ points, $\mathbf{c}_1, \ldots, \mathbf{c}_M$ and the origin.

We proposed an algorithm based on a random sampling. Our algorithm chooses a set of N weights $\{\mathbf{w}_n\}$ so that $\|\mathbf{w}_n\| = 1$ and their angles obey $U[0, \pi/2]$ independently. The nth weight $\mathbf{w}_n$ unifies the D objective functions to

$$L(\mathbf{s}; \mathbf{w}_n) = \sum_{i=1}^{D} (\mathbf{w}_n)_i f_i(\mathbf{x}), \tag{5}$$

which is maximized when $\mathbf{s} = \mathbf{s}(\mathbf{w}_n) \in C = \{\mathbf{c}_1, \ldots, \mathbf{c}_M\}$. Hence, the algorithm successfully reconstructs the feasible criterion space Z when $\{\mathbf{w}_n\}$ is chosen so that $\{\mathbf{s}(\mathbf{w}_n)\}_{n=1}^N = C$, and fails otherwise.

The problem of this paper is to derive the probability $\mathcal{E}$ that our algorithm fails to reconstruct Z from randomly chosen $\{\mathbf{w}_n\}$. More specifically, we derived an upper bound of its average $\mathrm{E}[\mathcal{E}]$ as shown in the sequel.

3 Result

The average of the error probability satisfies

$$\mathrm{E}\left[\mathcal{E}\right] \leq \binom{K}{D} \int_{B_+} p(B) dB V\left(\mathbf{f}_B\right)^{K-D} \left(1 - \frac{W(B)}{|S_+^{D-1}|}\right)^N \tag{6}$$

where $B = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_D\}$,

$$V(\mathbf{f}) = \int_{S_+^{D-1}} p(\omega) p\omega \int_{\omega^T \mathbf{f}}^{\infty} p(r) dr, \tag{7}$$

$$W(B) = \left| \left\{ \sum_{i=1}^{D} \lambda_i \mathbf{b}_k | \lambda_i > 0 \right\} \cap S_+^{D-1} \right|, \tag{8}$$

$$B_+ = \left\{ B | \mathbf{f}_B > 0 \right\}, \tag{9}$$

$$\mathbf{f}_B = \begin{bmatrix} \mathbf{b}_1^T \\ \cdots \\ \mathbf{b}_D^T \end{bmatrix}^{-1} \begin{bmatrix} \|\mathbf{b}_1\|^2 \\ \cdots \\ \|\mathbf{b}_D\|^2 \end{bmatrix}, \tag{10}$$

and $p(\omega)$ and $p(r)$ obey the uniform distribution on the $(D-1)$-dimensional sphere S_+^{D-1} and the χ^2 distribution with DOF k, respectively.

4 Proof Sketch

D hyperplanes intersect at one point with probability one. Without loss of generality, we consider the set B of D hyperplanes, $\mathbf{b}_1, \ldots, \mathbf{b}_D$. Then, the intersecting point $\mathbf{f}_B$ is expressed as (10) from (4).

The probability that our algorithm does not find $\mathbf{f}_B$ is the product of the probability of $\mathbf{f}_B \in C$ and that of $\{\mathbf{s}(\mathbf{w}_n)\}_{n=1}^N = C$ when $\mathbf{f}_{1 \cdots D} \in C$. We can calculate the two probabilities separately as below.

The intersecting point $\mathbf{f}_B$ of $B = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_D\}$ is included in C if and only if $\mathbf{f}_B > 0$ and $\mathbf{b}_k^T \mathbf{f}_B < \|\mathbf{b}_k\|^2$ for any $k (> D)$ (Fig. 4). Hence, the probability of $\mathbf{f}_B \in C$ is expressed as $V(\mathbf{f}_B)$ in (7).

The probability that a weight $\mathbf{w}$ at random finds $\mathbf{f}_B$, that is, $\mathbf{s}(\mathbf{w}) = \mathbf{f}_B$, is expressed as $W(B)/|S_+^{D-1}|$ since $W(B)$ in (8) represents the measure of $\mathbf{w}$ on S_+^{D-1} such that $\mathbf{s}(\mathbf{w}) = \mathbf{f}_B$ (Fig. 5). This measure can be calculated by considering a simplex in the hypersphere [11, 12].

Because the N weights are i.i.d., the probability of $\{\mathbf{s}(\mathbf{w}_n)\}_{n=1}^N = C$ when $\mathbf{f}_B \in C$ is given by

$$V\left(\mathbf{f}_B\right)^{K-D} \left(1 - \frac{W(B)}{|S_+^{D-1}|}\right)^N. \tag{11}$$

and the mean error probability is given by its average over B.

Since the number of $\mathbf{f}_B$'s is given by combinations, an upper bound of the mean error probability is given by (6).

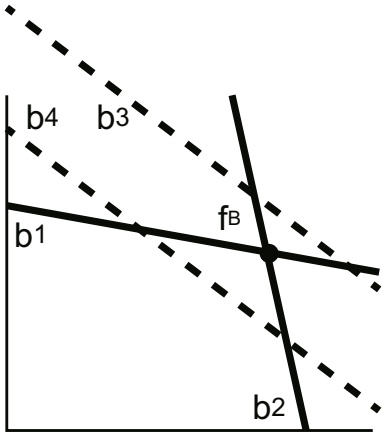


Fig. 4. An example of an intersecting point when $D = 2$ and $B = \{\mathbf{b}_1, \mathbf{b}_2\}$. $\mathbf{f}_B \in C$ when $\mathbf{b}_3$ is added while $\mathbf{f}_B \notin C$ when $\mathbf{b}_4$ is added.

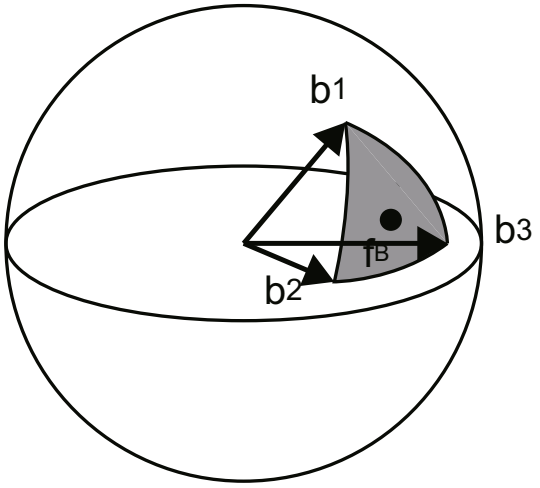


Fig. 5. The area of $\mathbf{w}$ such that $\mathbf{s}(\mathbf{w}) = \mathbf{f}_B$ when $D = 3$. The convex hull of $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ on S_+^{D-1}.

5 Conclusion

In this study, we proposed a method for reconstructing the Pareto set from a finite times of single-objective optimization. Our method is based on an idea similar to randomized algorithm or the PAC learning and works well with high probability. We also derived an upper bound of the mean error probability from the statistical viewpoint. The bound decreases as the number of sampling increases.

References

1. Marler, R.T., Arora, J.S.: Survey of Multi-Objective Optimization Methods for Engineering. Struct. Multidisc. Optim. 26, 369–395 (2004)
2. Heo, J.S., Lee, K.Y., Garduno-Ramirez, R.: Multiobjective control of power plants using particle swarm optimization techniques. IEEE Trans. Energy Conversion 21(2), 552–561 (2006)
3. Obayashi, S., Sasaki, D., Takeguchi, Y., Hirose, N.: Multiobjective evolutionary computation for supersonic wing-shape optimization. IEEE Trans. Evolutionary Computation 4(2), 182–187 (2000)
4. Das, I., Dennis, J.E.: Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. SIAM Journal on Optimization 8(3), 631–657 (1998)
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolutionary Computation 6(2), 182–197 (2002)
6. Durillo, J.J., García-Nieto, J., Nebro, A.J., Coello, C.A.C., Luna, F., Alba, E.: Multi-objective particle swarm optimizers: An experimental comparison. In: Ehrgott, M., Fonseca, C.M., Gandibleux, X., Hao, J.-K., Sevaux, M. (eds.) EMO 2009. LNCS, vol. 5467, pp. 495–509. Springer, Heidelberg (2009)
7. Solanki, R.S., Appino, P.A., Cohon, J.L.: Approximating the noninferior set in multiobjective linear programming problems. European Journal of Operational Research 68(3), 356–373 (1993)
8. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press and McGraw Hill (1990)
9. Valiant, L.G.: A theory of the learnable. Communications of ACM 27, 1134–1142 (1984)
10. Vapnik, V.N.: Statistical learning theory. John Wiley and Sons (1998)
11. Schläfli, L.: On the multiple integral $\int^n dxdy\cdots dz$ whose limits are $p_1 = a_1x + b_1y + \cdots + h_1z > 0, \ldots, p_n > 0$ and $x^2 + y^2 + \cdots + z^2 < 1$. The Quarterly Journal of Pure and Applied Mathematics 3, 54–68 (1860)
12. Kohno, T.: The volume of a hyperbolic simplex and iterated integrals. In Series on Knots and Everything. World Scientific (2007)

A Median Variant of Generalized Learning Vector Quantization

David Nebel[1,⋆], Barbara Hammer[2], and Thomas Villmann[1]

[1] Computational Intelligence Group,
University of Applied Sciences Mittweida, 09648 Mittweida, Germany
[2] Theoretical Computer Science Group,
University of Bielefeld, Germany

Abstract. We introduce a median variant of the Generalized Learning Vector Quantization (GLVQ) algorithm. Thus, GLVQ can be used for classification problem learning, for which only dissimilarity information between the objects to be classified is available. For this purpose, the cost function of GLVQ is reformulated as a probabilistic model such that a generalized expectation maximization scheme can be applied as learning procedure. We give a rigorous mathematical proof for the new approach. Exemplary examples demonstrate the performance and the behavior of the algorithm.

1 Introduction

Prototype based classification is one of the most successful paradigms in classification learning of vectorial data [14]. Prominent examples are support vector machines (SVM, [18]), Soft Nearest Prototype Classifier (SNPC,[20]) or Learning Vector Quantizers (LVQ,[15]). LVQ algorithms generate class typical prototypes whereas in SVMs the resulting prototypes determine the class borders and are here called support vectors. These support vectors are data points identified by convex optimization. Yet, LVQs as introduced by Kohonen realize a Hebbian learning but does not minimize a cost function. Sato&Yamada proposed a cost function based generalization of LVQ such that gradient descent learning can be applied (GLVQ,[17]). A probabilistic formulation of LVQ is the Robust Soft LVQ (RSLVQ, [21]).

If only dissimilarity information between the data is available, median and relational methods are required. Median methods restrict the prototypes to be data points whereas relational approaches allow prototypes to be linear combinations of data. Beginning with the pioneering work by J. Bezdek [1,13,12], median and relational variants of vector quantization are applied in unsupervised learning for clustering and data compression. Newer approaches also uses semi-supervised techniques [6,8]. Recently, a relational approach of the GLVQ was introduced assuming the prototypes are linear combinations of the data points

⋆ Supported by a grant of the European Social Foundation (ESF), Saxony.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 19–26, 2013.

[5,24]. However, a pure median variant of GLVQ is an open problem so far, which is addressed in this paper. For this purpose, we reformulate the cost function of GLVQ in terms of a *probabilistic* model, such that a variant of the expectation maximization (EM) strategy becomes applicable. Doing so, generalized EM optimization yields the desired functionality.

The paper is structured as follows: First we briefly review the standard definition of GLVQ. Second, we give a reformulation suitable for an EM variant. We provide a mathematical proof for this algorithm. Finally we demonstrate the approach for exemplary applications and summarize with concluding remarks.

2 A Median Variant of the Generalized Learning Vector Quantization Algorithm

Learning vector quantization comprises a family of prototype based vector quantizers for classification of vectorial data, which are trained according to a heuristic to minimize the classification error [15]. After learning unknown data points are classified to that class, the closest prototype is belonging. The GLVQ algorithm is a generalization of the heuristic learning scheme formalizing the objective to optimize the hypothesis margin of the classifier [4,10,19]. It approximates the classification error by a *differentiable* cost function such that gradient descent learning becomes available [17].

2.1 The Cost Function of GLVQ

Let $x_i \in \mathbb{X}\ i = 1, .., N$ be the data points to be learned and $w_j \in \mathbb{W}\ j = 1, .., M$ be the prototypes. Further, let $c(\cdot)$ be the formal class label function, which assigns to each data point the class label $c\,(x_i)$. Analogously, $c\,(w_j)$ returns the class label of the prototype. We introduce the distances $d^+(x_i)$ and $d^-(x_i)$ as

$$d^+(x_i) = \min_{\{w_j : c(x_i) = c(w_j)\}} d(x_i, w_j)$$
$$d^-(x_i) = \min_{\{w_j : c(x_i) \neq c(w_j)\}} d(x_i, w_j)$$

describing the minimal distances from x_i to the closest prototype of the same class (correct) and to closest prototype of any other class (incorrect), respectively. The classifier function

$$\mu_\alpha(x_i) = \frac{d^-(x_i) - d^+(x_i)}{d^+(x_i) + d^-(x_i)} + \alpha \tag{1}$$

with $\mu_\alpha(x_i) \in I_\alpha = [-1 + \alpha, 1 + \alpha]$. For $\alpha = 0$, the classifier function $\mu_\alpha(x_i)$ becomes negative if $d^+(x_i) > d^-(x_i)$ is valid, i.e. data point would be incorrectly classified. An value $\alpha \neq 0$ would shift this decision boundary as well as the interval I_α. Sato&Yamada defined the cost function of GLVQ to be minimized by stochastic gradient descent learning as

$$C(\mathbb{X}, \mathbb{W}) = \sum_{i=1}^{N} f(-\mu_\alpha(x_i)) \tag{2}$$

where f is a monotonically increasing transfer function (e.g. the identity function) [17]. In fact, the choice of α does not influences the optimization.

To meet the requirements for the *generalized* EM-learning we translate the minimization problem 2) into a maximization problem

$$K(\mathbb{X}, \mathbb{W}) = \sum_{i=1}^{N} \log \left(g^+(x_i, \mathbb{W}) + g^-(x_i, \mathbb{W}) \right) \tag{3}$$

specifying the transfer function as the logarithm and introducing the *positive* quantities

$$g^+(x_i, \mathbb{W}) = \frac{\alpha}{2} - \frac{d^+(x_i)}{d^+(x_i) + d^-(x_i)} \tag{4}$$

$$g^-(x_i, \mathbb{W}) = \frac{\alpha}{2} + \frac{d^-(x_i)}{d^+(x_i) + d^-(x_i)}. \tag{5}$$

This choice implies $a > 1$ to ensure $\mu_\alpha(x_i) > 0$. for suitably chosen, we obtain an equivalent optimization function for GLVQ now to be maximized. From numerical point of view $g^\pm \geq 1$ should be valid to avoid instabilities from the logarithm in case of small values. One easily checks that $\alpha \geq 4$ fulfills this requirement.

2.2 Reformulation of the GLVQ Cost Function and Generalized EM Optimization

In the following we reformulate the modified cost function $K(\mathbb{X}, \mathbb{W})$ from (3) for application of a generalized EM strategy (gEM). For this reason we introduce the *formal probabilities*

$$p^+(\mathbb{W}|x_i) = \frac{g^+(x_i, \mathbb{W})}{g^+(x_i, \mathbb{W}) + g^-(x_i, \mathbb{W})} \tag{6}$$

$$p^-(\mathbb{W}|x_i) = \frac{g^-(x_i, \mathbb{W})}{g^+(x_i, \mathbb{W}) + g^-(x_i, \mathbb{W})} \tag{7}$$

which sum up to $p^+(\mathbb{W}|x_i) + p^-(\mathbb{W}|x_i) = 1$. Hence, both probabilities form a probability density $p(\mathbb{W}|x_i)$ for a given data point x_i. Additionally, we introduce the functions $\gamma^+(\mathbb{W}|x_i) \geq 0$ and $\gamma^-(\mathbb{W}|x_i) \geq 0$, which play the role of generating models for the prototypes for correct and incorrect classification of a given data point x_i. We consider the additional constraint

$$\gamma^+(\mathbb{W}|x_i) + \gamma^-(\mathbb{W}|x_i) = 1 \tag{8}$$

such that both, $\gamma^+(\mathbb{W}|x_i)$ and $\gamma^-(\mathbb{W}|x_i)$, form together a *formal probability density function* $\gamma(\mathbb{W}|x_i)$. The respective Kullback-Leibler-divergence (KLD) is calculated as

$$\mathcal{K}_i(\gamma||p) = \gamma^+(\mathbb{W}|x_i) \cdot \log \left(\frac{p^+(\mathbb{W}|x_i)}{\gamma^+(\mathbb{W}|x_i)} \right) + \gamma^-(\mathbb{W}|x_i) \cdot \log \left(\frac{p^-(\mathbb{W}|x_i)}{\gamma^-(\mathbb{W}|x_i)} \right) \tag{9}$$

and, hence, $\mathcal{K}_i(\gamma||p) \geq 0$ holds for each x_i.

Further, we can formally introduce the *generalized* KLD (see [3,23])

$$\mathcal{L}_i(\gamma||g) = \gamma^+(\mathbb{W}|x_i) \cdot \log\left(\frac{g^+(x_i,\mathbb{W})}{\gamma^+(\mathbb{W}|x_i)}\right) + \gamma^-(\mathbb{W}|x_i) \cdot \log\left(\frac{g^-(x_i,\mathbb{W})}{\gamma^-(\mathbb{W}|x_i)}\right) \quad (10)$$

with $g = \{g^+(x_i,\mathbb{W}), g^-(x_i,\mathbb{W})\}$. Now, we can decompose (3) into

$$K(\mathbb{X},\mathbb{W}) = \sum_{i=1}^{N} \left[\mathcal{L}_i(\gamma||g) - \mathcal{K}_i(\gamma||p)\right]. \quad (11)$$

which is, in fact, has the structure of a maximum likelihood problem. The first term $\mathcal{L} = \sum_{i=1}^{N} \mathcal{L}_i(\gamma,\Theta)$ is a lower bound for the cost function because the divergence property of the $\mathcal{K}_i(\gamma||p)$. The proof of the decomposition is given in the Appendix.

With the decomposition (11) and the properties described above, we can specify an EM optimization approach for finding a maximum of the likelihood function, see Alg. 1.

Algorithm 1. gEM algorithm of GLVQ

1. Initialize $\mathbb{W}^{\text{old}}$
2. **E Step:** set $\gamma(\mathbb{W}|x_i) \leftarrow p(\mathbb{W}^{\text{old}}|x_i)$
3. **M Step:** for fixed $\gamma(\mathbb{W}|x_i)$ determine $\mathbb{W}^{\text{new}} = \arg\max_{\mathbb{W}} \sum_{i=1}^{N} \mathcal{L}_i(\gamma||g)$, which improves $\mathcal{L}$
4. If $\mathbb{W}^{\text{new}} = \mathbb{W}^{\text{old}}$ then STOP, else set: $\mathbb{W}^{\text{new}} \leftarrow \mathbb{W}^{\text{old}}$ and go to step 2.

Remark 1. At this point we emphasize that for this variant of the gEM-approach we do not search for the set $\mathbb{W}^*$ in the M-step, which would maximize the cost function $K(\mathbb{X},\mathbb{W})$. We only assume that the cost function is not decreasing for $\mathbb{W}^{\text{new}}$.

Obviously, the algorithm only requires the distances $d(x_i, w_j)$ between data and prototypes. If we restrict the prototypes to be data points, which corresponds exactly to the *median* principle, only the dissimilarities between the data are needed. Hence, the gEM formulation of GLVQ delivers a median variant (mGLVQ). This also allows the application of non-standard metrics or dissimilarities like divergences, correlations, kernels etc. for dissimilarity judgment of data [11,22,23]. Further, structured data with non-standard metric like edit distances for text sequences or mixed dissimilarities for heterogeneous data can be investigated [7,25].

Now we give the proof for consistency of the Alg. 1:

Proof. For mathematical consistency of the gEM-approach for GLVQ it remains to show that each iteration step is non-decreasing with respect to the cost function $K(\mathbb{X},\mathbb{W})$: In the E-Step we only change the density function $\gamma(\mathbb{W}|x_i)$ but not the prototype set $\mathbb{W}$. The cost function itself does not depend on $\gamma(\mathbb{W}|x_i)$ and, therefore, remains unchanged. On the other hand, the

E-step implies $\sum_{i=1}^{N} \mathcal{K}_i(\gamma||p) = 0$. Consequently, the E-step increases the lower bound value $\mathcal{L}$ and, hence, $K(\mathbb{X}, \Theta) = \mathcal{L}$ holds at this time.

In the M-step, we take the densities $\gamma(\mathbb{W}|x_i)$ fixed, and therefore, they are independent from $\mathbb{W}$. We maximize the lower bound $\mathcal{L}$ with respect to $\mathbb{W}$. Consequently, we have no reduction in the value of the cost function. Yet, this change in the cost function can be larger than the change of the lower bound, because the divergences $\mathcal{K}_i(\gamma||p)$ also depends on $\mathbb{W}$ and may contribute. Thus, the cost function is non-decreasing during the iteration procedure, which proofs the consistency of the proposed approach.

3 Experiments

We evaluate the proposed mGLVQ-model in comparison to alternatives using the benchmark scenarios as proposed in [2]. These benchmarks contain dissimilarity data represented in terms of pairwise dissimilarities only. In general, these data are non-Euclidean, such that SVM techniques cannot directly be applied. The approach [2] investigates a preprocessing of the data by diverse techniques to enforce a positive semi-definite kernel for SVM. In addition to SVM, we compare to kernel LVQ variants and relational LVQ, which implicitly embed data in Euclidean or pseudo-Euclidean space [24]. For SVM and kernel variants, preprocessing of non-Euclidean data is necessary; for this purpose the best results obtained by clip, flip, or shift are reported [2].

The data sets used are as follows [2]:

1. *Voting* contains 435 samples in 2 classes, representing categorical data compared based on the value difference metric.
2. *Aural Sonar* consists of 100 signals with two classes (target of interest/clutter), representing sonar signals with dissimilarity measures according to an ad hoc classification of humans.
3. *Protein* consists of 213 data from 4 classes, representing globin proteins compared by an evolutionary measure.
4. *Face Recognition* consists of 945 samples with 139 classes, representing faces of people, compared by the cosine similarity.

All data sets are characterized by symmetric dissimilarity matrices only, which are generally not Euclidean. The non-Euclideanity is judged by the *signature* Σ, which corresponds to the triplet formed by the number of positive, negative, and (numerically) zero eigenvalues of a pseudo-Euclidean embedding of the data, respectively [16]: The data are Euclidean iff the second entry of Σ is zero. For the data as described above, we obtain the following signature values Σ:

data set	Voting	Aural	Protein	FaceRec
Σ	(16,1,418)	(61,38,1)	(169,38,6)	(45,0,900)

This indicates, that *Voting* and *FaceRec* are almost Euclidean while the other data contain a significant contribution of non-Euclidean nature.

For all experiments, the setup as described in [2] was used, i.e. results are obtained by a repeated ten-fold cross-validation with ten repeats. Parameters are optimized by a cross-validation within this scheme. The number of prototypes is chosen as a small multiple of the number of classes with random initialization. We report the result of mGLVQ, which can be derived in an analogous way based on the GLVQ cost function, the latter implicitly formalizing the objective to optimize the hypothesis margin of the classifier [4,10,19]. To avoid local optima while iterative optimization of the M-step, we use 10 random restarts for this step.

Table 1. Results of Median GLVQ (mGLVQ) in comparison with the best results for kernel GLVQ (kGLVQ,[9]), and Support Vector Machines (SVM,[18]) taking the best data preprocessing from clip/flip/shift for SVM and kGLVQ. The classification accuracies were produced by repeated 10-fold cross-validation with 10 repeats. The last column contains the number of prototypes used for mGLVQ and in brackets the number of prototypes which was used for the kGLVQ.

	mGLVQ	kGLVQ	SVM	# Prototypes
Voting	**0.956**	0.9466	0.9511	20 (20)
Aural	**0.907**	0.8875	0.88	6 (10)
Protein	0.904	**0.986**	0.9802	4 (20)
Face Rec	**0.987**	0.9665	0.9627	139 (139)

In all but one case (Protein), the results obtained by mGLVQ are comparable to best results obtained by SVM and kernel GLVQ (kGLVQ,[9]), which implicitly embed the data in a high dimensional Hilbert space (possibly after preprocessing a non-Euclidean data matrix), or pseudo-Euclidean case, respectively. Further, mGLVQ represents prototypes in the form of a *single* exemplar, i.e. a data point, which can be directly inspected by a human observer in the same form as data points. This is in difference to SVM and kGLVQ, which represent prototypes in a distributed way, i.e. in terms of a small number of representative exemplars.

4 Conclusion

The GLVQ model has been extended to general dissimilarity data by means of reformulation of the GLVQ cost function in terms of a probabilistic model. Optimization of this new cost function can be done by a generalized EM scheme, while preserving the intuitive interpretability of classical LVQ. We give a mathematical proof for the convergence of the algorithm. The GLVQ becomes available for general dissimilarity data. In case of prototypes restricted to be data points, a median variant is obtained (mGLVQ).

References

[1] Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)

[2] Chen, Y., Garcia, E.K., Gupta, M.R., Rahimi, A., Cazzanti, L.: Similarity-based classification: Concepts and algorithms. Journal of Machine Learning Research 10, 747–776 (2009)

[3] Cichocki, A., Zdunek, R., Phan, A.H., Amari, S.-I.: Nonnegative Matrix and Tensor Factorizations. Wiley, Chichester (2009)
[4] Crammer, K., Gilad-Bachrach, R., Navot, A., Tishby, A.: Margin analysis of the LVQ algorithm. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing (Proc. NIPS 2002), vol. 15, pp. 462–469. MIT Press, Cambridge (2003)
[5] Gisbrecht, A., Mokbel, B., Schleif, F.-M., Zhu, X., Hammer, B.: Linear time relational prototype based learning. International Journal of Neural Systems 22(5), 1250021 (2012)
[6] Hammer, B., Hasenfuss, A., Schleif, F.-M., Villmann, T.: Supervised median clustering. IfI Technical Report Series IfI-09-06, TU Clausthal, Department of Informatics, Clausthal-Zellerfeld, Germany (2009)
[7] Hammer, B., Micheli, A., Strickert, M., Sperduti, A.: A general framework for unsupervised processing of structured data. Neurocomputing (2004) (page in press)
[8] Hammer, B., Hasenfuss, A.: Topographic mapping of large dissimilarity data sets. Neural Computation 22(9), 2229–2284 (2010)
[9] Hammer, B., Hofmann, D., Schleif, F.-M., Zhu, X.: Learning vector quantization for (dis-)similarities. Neurocomputing (page in press, 2013)
[10] Hammer, B., Strickert, M., Villmann, T.: Relevance LVQ versus SVM. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 592–597. Springer, Heidelberg (2004)
[11] Hammer, B., Strickert, M., Villmann, T.: Supervised neural gas with general similarity measure. Neural Processing Letters 21(1), 21–44 (2005)
[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 clustering algorithms. Pattern Recognition 22(3), 205–212 (1989)
[14] Haykin, S.: Neural Networks - A Comprehensive Foundation. IEEE Press, New York (1994)
[15] Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 30. Springer, Heidelberg (1995) (Second Extended Edition1997)
[16] Pekalska, E., Duin, R.P.W.: The Dissimilarity Representation for Pattern Recognition: Foundations and Applications. World Scientific (2006)
[17] Sato, A., Yamada, K.: Generalized learning vector quantization. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) Proceedings of the 1995 Conference on Advances in Neural Information Processing Systems, vol. 8, pp. 423–429. MIT Press, Cambridge (1996)
[18] Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press (2002)
[19] Schneider, P., Hammer, B., Biehl, M.: Distance learning in discriminative vector quantization. Neural Computation 21, 2942–2969 (2009)
[20] Seo, S., Bode, M., Obermayer, K.: Soft nearest prototype classification. IEEE Transaction on Neural Networks 14, 390–398 (2003)
[21] Seo, S., Obermayer, K.: Soft learning vector quantization. Neural Computation 15, 1589–1604 (2003)
[22] Strickert, M., Schleif, F.-M., Seiffert, U., Villmann, T.: Derivatives of Pearson correlation for gradient-based analysis of biomedical data. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial (37), 37–44 (2008)
[23] Villmann, T., Haase, S.: Divergence based vector quantization. Neural Computation 23(5), 1343–1392 (2011)

[24] Zhu, X., Schleif, F.-M., Hammer, B.: Semi-supervised vector quantization for proximity data. In: Verleysen, M. (ed.) Proc. of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2013), Louvain-La-Neuve, Belgium, pp. 89–94 (2013), i6doc.com

[25] Zühlke, D., Schleif, F.-M., Geweniger, T., Haase, S., Villmann, T.: Learning vector quantization for heterogeneous structured data. In: Verleysen, M. (ed.) Proc. of European Symposium on Artificial Neural Networks (ESANN 2010), pp. 271–276. d-side publications, Evere (2010)

Appendix - Proof That Equation (11) Is Valid

Proof.

$$
\begin{aligned}
\sum_{i=1}^{N}\left[\mathcal{L}_i(\gamma,\mathbb{W})-\mathcal{K}_i(\gamma||p)\right] &= \sum_{i=1}^{N}\Bigg[\gamma(\mathbb{W}|x_i)\log\left(\frac{g^+(x_i,\mathbb{W})}{\gamma^+(\mathbb{W}|x_i)}\right)+\gamma^-(\mathbb{W}|x_i)\log\left(\frac{g^-(x_i,\mathbb{W})}{\gamma^-(\mathbb{W}|x_i)}\right)\\
&\quad -\gamma^+(\mathbb{W}|x_i)\log\left(\frac{p^+(\mathbb{W}|x_i)}{\gamma^+(\mathbb{W}|x_i)}\right)-\gamma^-(\mathbb{W}|x_i)\log\left(\frac{p^-(\mathbb{W}|x_i)}{\gamma^-(\mathbb{W}|x_i)}\right)\Bigg]\\
&= \sum_{i=1}^{N}\Bigg[\gamma(\mathbb{W}|x_i)\log\left(\frac{g^+(x_i,\mathbb{W})}{\gamma^+(\mathbb{W}|x_i)}\right)+\gamma^-(\mathbb{W}|x_i)\log\left(\frac{g^-(x_i,\mathbb{W})}{\gamma^-(\mathbb{W}|x_i)}\right)\\
&\quad -\gamma^+(\mathbb{W}|x_i)\log\left(\frac{g^+(x_i,\mathbb{W})}{\gamma^+(\mathbb{W}|x_i)\left(g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})\right)}\right)\\
&\quad -\gamma^-(\mathbb{W}|x_i)\log\left(\frac{g^-(x_i,\mathbb{W})}{\gamma^-(\mathbb{W}|x_i)\left(g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})\right)}\right)\Bigg]\\
&= -\sum_{i=1}^{N}\Bigg[\gamma^+(\mathbb{W}|x_i)\log\left(\frac{1}{g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})}\right)\\
&\quad +\gamma^-(\mathbb{W}|x_i)\log\left(\frac{1}{g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})}\right)\Bigg]\\
&= \sum_{i=1}^{N}\Bigg[\gamma^+(\mathbb{W}|x_i)\log\left(g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})\right)\\
&\quad \gamma^-(\mathbb{W}|x_i)\log\left(g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})\right)\Bigg]\\
&= \sum_{i=1}^{N}\Bigg[\underbrace{\left(\gamma^+(\mathbb{W}|x_i)+\gamma^-(\mathbb{W}|x_i)\right)}_{=1}\log\left(g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})\right)\Bigg]\\
&= \sum_{i=1}^{N}\log\left(g^+(x_i,\mathbb{W})+g^-(x_i,\mathbb{W})\right)\\
&= K(\mathbb{X},\mathbb{W})
\end{aligned}
$$

MISUP: Multiple-Instance Learning via Immunological Suppression Mechanism

Duzhou Zhang[1,2] and Xibin Cao[1,⋆]

[1] School of Astronautics, Harbin Institute of Technology
[2] National Key Laboratory for Space Intelligent Control Technology, China Academy of Space Technology
{dzzhang_hit,xbcao_hit}@163.com

Abstract. In multiple-instance learning (MIL), examples are sets of instances named bags and labels are associated with bags rather than instances. A bag is labeled as positive if it contains at least one positive instance; otherwise, labeled as negative. Recently, several instance selection-based MIL (ISMIL) algorithms show their power in solving the MIL problem. In this paper, we propose a new ISMIL algorithm based on the self-regulation and suppression mechanisms found in the biological immune system. Experimental results show that our MIL algorithm is highly comparable with other ISMIL ones in terms of classification accuracy and computation time.

Keywords: Multiple-instance learning, Instance selection, Artificial immune systems, Support vector machines.

1 Introduction

Multiple-instance learning (MIL) is a variation on standard supervised learning, which was first introduced by Dietterich et al. when they were investigating the problem of drug activity prediction [1]. In this learning framework, training examples are bags of instances not single instances. Labels are associated with bags rather than instances in bags. A bag is labeled as positive if it contains at least one positive instance; otherwise, labeled as negative. The aim of a MIL algorithm is to learn a classifier for predicting the labels of unseen bags. The notion of bag together with the labeling protocol often make MIL more realistic than standard supervised learning for particular types of applications, such as drug activity prediction [1], stock selection [2], computer aided diagnosis [3] and content-based image retrieval (CBIR) [4].

In recent years, several instance selection-based MIL (ISMIL) algorithms have been proposed, including DD-SVM [7], MILES [8], MILD [9] and MILIS [10], which will be described later in Sect. 2. Although these algorithms convey competitive performance compared to the state-of-the-art MIL ones, they usually require lots of computation time to complete the learning task, especially for

⋆ Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 27–34, 2013.

large-scale datasets. As we know, computational efficiency is an important issue for practical applications, so it is necessary to design an efficient instance selection strategy to speed up the learning process while not sacrificing generalization accuracy much. In this paper, we propose a novel MIL algorithm based on an efficient instance selection strategy, which is inspired by the self-regulation and suppression mechanisms found in the biological immune system. We call it Multiple-Instance Learning via Immunological Suppression Mechanism (MISUP). Due to our instance selection strategy, MISUP could generate highly comparative classification results as compared to other ISMIL algorithms. Meanwhile, MISUP could complete the process of instance selection more efficiently, and thus it is more efficient in terms of computation time.

The remainder of this paper is organized as follows. In Sect. 2, we review some related work to our research. In Sect. 3, we present MISUP. In Sect. 4, we evaluate MISUP on two MIL tasks, i.e. drug activity prediction and region-based image categorization. Finally, we conclude in Sect. 5.

2 Related Work

Since MIL was first proposed in the context of drug activity prediction, many efforts have been endeavored to address this learning paradigm. The first MIL algorithm is called axis-parallel rectangle (APR), which is aimed at finding an APR including at least one instance from each positive bag but excluding all instances from negative bags [1]. A bag is classified as positive if at least one of its instances falls within the APR; otherwise, it is classified as negative. Similarly, Maron and Lozano-Pérez proposed a new concept called diverse density (DD) for MIL, which measures how many different positive bags have instances near a point in the feature space and how far negative instances are from that point [2]. The EM-DD algorithm combines DD with expectation maximization, aimed at locating the target concept in a more efficient manner [11]. The mi-SVM/MI-SVM algorithm treats the unobservable instance labels as hidden variables and formulates MIL as a mixed integer quadratic program [12].

Recently, several ISMIL algorithms have been presented, namely DD-SVM [7], MILES [8], MILD [9] and MILIS [10]. The basic idea is mapping each bag into a new feature space called embedding space, which is constructed using some instance prototypes chosen from the training set. Thus, training bags are represented by single feature vectors and the MIL problem is converted to a standard supervised learning one. Then standard SVMs are trained using these bag-level feature vectors. Specifically, DD-SVM uses the DD function [2] to identify instance prototypes. MILES considers all instances in the training set as initial instance prototypes and instance selection is implicitly performed via learning a 1-norm SVM with a linear kernel. MILD performs instance selection based on a conditional probability model. MILIS achieves the initial instance selection by modeling the distribution of the negative population with the Gaussian-kernel-based kernel density estimator. Then it depends on an iterative optimization framework to update instance prototypes and learn a linear SVM.

3 MISUP: Multiple-Instance Learning via Immunological Suppression Mechanism

3.1 Self-Regulation and Suppression Mechanisms

Artificial immune systems (AIS) constitute a relatively new area of bio-inspired computing. Biological models of natural immune systems have provided the inspiration for AIS algorithms [13]. A biological immune system is composed of diverse sets of cells and molecules that work together with other systems, such as the neural and endocrine systems, in order to maintain a homeostatic state within the host. The traditionally held view of the role of the immune system is to protect our bodies from invading infectious agents known as *pathogens*, and an immune response to a pathogen is provoked by the recognition of an associated molecule called *antigen*. There are two arms of the immune system: innate and adaptive [14]. Innate immunity is not directed towards specific invaders but against general pathogens that enter the body [15], while adaptive immunity allows the immune system to launch an attack against any invader that the innate system cannot remove [16]. The adaptive immune system mainly consists of lymphocytes, which are white blood cells (*WBCs*), more specifically B and T cells. These cells aid in the process of recognising and destroying specific substances.

During immune response, the most successful WBCs in mounting the immune response to pathogens receive stimulus to proliferate. The least effective ones are eliminated from the organism due to one of the immune system's characteristics, i.e. self-regulation mechanism. In the self-regulation mechanism, immune cells that are no longer needed by the organism do not receive signals to stay alive and thus they die [16]. Those signals come from the lymph nodes or other helper cells. Moreover, suppressive signals also balance different types of pathogen-specific WBCs according to the infection inside the organism, favoring the proliferation of those immune cells most needed for defense in a certain point. Once the infection is controlled, the excessive pathogen-specific WBCs are eliminated. The self-regulation and suppression mechanisms allow the organism to save energy and keep only the WBC repertoire most needed for self defense.

3.2 The Proposed Algorithm

Inspired by the self-regulation and suppression mechanisms, Figueredo et al. proposed a fast immune-inspired suppressive algorithm for instance selection in supervised learning, namely SeleSup [17]. According to the self-regulation mechanism, those cells unable to neutralize danger tend to disappear from the organism (or to be suppressed). By analogy, data not relevant to the learning of a classifier are eliminated from the training process. SeleSup takes advantage of the suppression concept and applies it to the training process in order to eliminate very similar data instances and keep only the representative ones.

To perform such tasks, SeleSup divides the dataset into two subsets. The first one represents the WBCs (training set) in the organism. The second one represents a set of pathogens (suppression set) that is responsible for selecting

the higher affinity WBCs, i.e. conducting the suppression. The algorithm starts with the idea that the system's model must identify the best subset of WBCs in order to recognize pathogens. Specifically, each pathogen in the suppression set is classified according to the closest WBC in the training set. Those WBCs able to recognize pathogens remain while others are eliminated. This recognition ability is obtained via comparing the label of the closest WBC with that of the corresponding pathogen. The closest WBC is considered having the ability to recognize the corresponding pathogen if their labels are the same, and lacking this ability otherwise. Note that the Euclidean distance is adopted to measure the affinity between a WBC and a pathogen in the SeleSup algorithm.

The self-regulation and suppression mechanisms used by SeleSup and its major process remain in our MISUP algorithm. However, examples are bags composed of one or more instances in MIL, the new distance metric has to be adopted to evaluate the affinity between a WBC and a pathogen. For the current investigation, the minimal Hausdorff distance [18] is adopted for this purpose, since it has been successfully applied in many MIL problems. Formally, given two bags B_i and B_j, the minimal Hausdorff distance between them is defined as

$$\mathrm{H}_{min}(B_i, B_j) = \min_{B_{it} \in B_i, B_{js} \in B_j} \|B_{it} - B_{js}\| \ , \tag{1}$$

where $\|B_{it} - B_{js}\|$ measures the distance between two instances B_{it} and B_{js}, which takes the form of Euclidean distance here. It should be noted here that while the affinity measure for the current work relies solely on the minimal Hausdorff distance, other alternatives of set-based distance metrics may be viable.

From the definition of the minimal Hausdorff distance in (1), we know that the distance between two bags is determined by the closest two instances from them. Thus, the closest instances between a WBC and a pathogen play a key role in evaluating the relation of them. Further, the closest instance from a WBC (relative to its closest instance in the corresponding pathogen) and the WBC's label determine if this WBC is kept in the final WBC repertoire. Inspired by this observation, we attempt to use the closest instance from a WBC with the same label as its relevant pathogen as an instance prototype. The pseudo-code for instance selection in MISUP has been summarized in Algorithm 1.

When the whole process in Algorithm 1 finishes, a set of instance prototypes remains. Like other ISMIL algorithms, all such instance prototypes form the embedding space. Now, only one issue has to be addressed, i.e. the definition of bag-level feature mapping. As in DD-SVM and MILD, we use a similar feature mapping to map every bag B_i to a point $\lambda(B_i)$ in the embedding space as follows:

$$\lambda(B_i) = [d(B_i, t_1), d(B_i, t_2), \ldots, d(B_i, t_n)]^{\mathrm{T}} \ , \tag{2}$$

where $d(\cdot, \cdot)$ measures the minimal Hausdorff distance between a bag and an instance prototype, and $t_k \in T$, and T is the set of instance prototypes, and n is the size of T. Thus, training bags are represented by single bag-level feature vectors using (2). Then a standard SVM with a Gaussian kernel is trained using these bag-level feature vectors.

Algorithm 1. Pseudo-code for instance selection in MISUP

Input: Full training set B and the fraction f of WBCs
Output: Set of instance prototypes T

1: Randomly assign $\lceil f \cdot |B| \rceil$ examples as **WBCs** (training set); The remaining examples are assigned as **Pathogens** (suppression set)
2: Set a survival signal for every instance in every WBC and initialize it to be false
3: **for** p in **Pathogens do**
4: $NearestWBC = \arg\min_{w \in \mathbf{WBCs}} \mathrm{H}_{min}(p, w)$
5: Find the closest instance t in $NearestWBC$ with respect to p
6: **if** $NearestWBC.label == p.label$ & $t.survival ==$ false **then**
7: add t to T
8: $t.survival =$ true

4 Experiments and Analysis

4.1 Drug Activity Prediction

The MUSK datasets, MUSK1 and MUSK2, are standard benchmark datasets for MIL [1], which are publicly available from the UCI Machine Learning Repository [19]. These datasets consist of descriptions of molecules and the task is to predict whether a given molecule is active or inactive. Each molecule is viewed as a bag, the instances of which are the different low-energy conformations of the molecule. If one of the conformations of a molecule binds well to the target protein, the molecule is said active, and inactive otherwise. MUSK1 contains 47 positive bags and 45 negative bags. MUSK2 contains 39 positive bags and 63 negative bags. MUSK2 shares 72 molecules with MUSK1, but includes more conformations for those shared molecules.

We used LIBSVM [20] to train all the SVMs. The regularization parameter C and the Gaussian kernel parameter γ need to be specified for MISUP. These parameters were selected according to a twofold cross-validation on the training set. Both C and γ were chosen from $\{2^{-10}, 2^{-8}, \ldots, 2^{10}\}$ and the pair of values giving the minimum twofold cross-validation error was chosen to set these two parameters. We found that $C = 2^0$ and $\gamma = 2^0$ gave the minimum twofold cross-validation error on MUSK1, and $C = 2^8$ and $\gamma = 2^{-6}$ on MUSK2. The fraction of WBCs f in Algorithm 1 was set to 0.5. We fixed these values for the subsequent experiments. As for DD-SVM [7], MILES [8], MILD [9] and MILIS [10], we used the same setting to determine the parameters required by them.

Table 1 reports the mean and 95% confidence interval of the results over ten runs of tenfold cross-validation. Moreover, we have tested other ISMIL algorithms. For completeness, we have also included the results from some other MIL algorithms. Table 1 shows that APR and MILES achieve the best performance on MUSK1 and MUSK2 datasets, respectively. Our MISUP algorithm is highly comparable with other ones. In terms of the average prediction accuracy over the two datasets, MISUP is the third best among all those algorithms listed in Table 1 (90.8% for APR, 89.2% for MILES, 88.1% for MISUP). It is noteworthy that MISUP is the second best among all the ISMIL algorithms (DD-SVM,

Table 1. Classification accuracies (%) of various algorithms on the MUSK datasets

Algorithm	MUSK1	MUSK2
MISUP	87.7 : [87.0, 88.4]	88.5 : [87.1, 89.8]
DD-SVM [7]	78.6 : [77.1, 80.1]	85.3 : [84.5, 86.1]
MILES [8]	87.4 : [86.1, 88.7]	90.9 : [90.0, 91.8]
MILD [9]	82.2 : [80.8, 83.6]	86.2 : [84.5, 87.9]
MILIS [10]	84.4 : [83.2, 85.6]	88.8 : [87.6, 90.0]
APR [1]	92.4	89.2
DD [2]	88.9	82.5
EM-DD [11]	84.8	84.9
MI-SVM [12]	77.9	84.3
mi-SVM [12]	87.4	83.6

Table 2. Computation time (minutes) of all ISMIL algorithms on the MUSK datasets: time spent on model selection + training time after model selection

Algorithm	MUSK1	MUSK2
MISUP	0.2 + 0.2	14.6 + 17.7
DD-SVM [7]	165.4 + 111.5	2321.8 + 1170.1
MILES [8]	1.0 + 1.0	180.8 + 218.9
MILD [9]	0.5 + 0.4	36.6 + 27.4
MILIS [10]	2.5 + 2.0	705.9 + 814.2

MILES, MILD and MILIS) with respect to the overall prediction accuracy on the MUSK datasets. Meanwhile, the difference between the accuracy of MISUP and the highest accuracy of MILES is only 1.1%.

The we evaluate MISUP with respect to computational efficiency. Table 2 reports the overall computation time required by various ISMIL algorithms on the MUSK datasets. The time spent on model selection and the training time after model selection are reported separately. The time spent on model selection is that consumed on selecting the optimal parameter values for every ISMIL algorithm. The training time after model selection is the total training time of ten runs of tenfold cross-validation. All the experiments were conducted on a 3.1 GHz PC. We can find that MISUP is much more efficient than other algorithms. The speedup of MISUP over other algorithms for the MUSK2 dataset is more obvious due to the large number of instances in this dataset.

4.2 Region-Based Image Categorization

The COREL dataset has been widely used for region-based image categorization. The dataset contains 20 thematically diverse image categories with 100 images of size 384×256 or 256×384 in each category. Each image is segmented into several local regions and features are extracted from each region. The dataset and extracted features are available at `http://www.cs.olemiss.edu/~ychen/ddsvm.html`.

Table 3. Classification accuracies (%) of all ISMIL algorithms on the COREL datasets

Algorithm	COREL_{10}	COREL_{20}
MISUP	80.5 : [79.2, 81.9]	69.7 : [68.2, 71.3]
DD-SVM [7]	80.2 : [79.5, 80.9]	67.8 : [66.7, 68.9]
MILES [8]	82.3 : [81.6, 83.0]	71.7 : [70.8, 72.6]
MILD [9]	76.2 : [75.3, 77.1]	68.4 : [67.4, 69.4]
MILIS [10]	82.4 : [81.8, 83.0]	69.6 : [68.7, 70.5]

Since this is a multiclass classification problem, we apply the one-against-the-rest approach to train 20 binary SVMs. A test bag is assigned to the category with the largest decision value given by the SVMs.

We have conducted two tests for the 10-category and 20-category categorizations. The first 10 categories in the COREL dataset were used in the first test while all 20 categories were used in the second test. For each category, we randomly selected half of images as training bags and the remaining half as test bags. Training and testing were repeated for five different random partitions. We used the same experimental setting as in Sect. 4.1 to determine the regularization parameter C and the Gaussian kernel parameter γ for MISUP, as well as the corresponding parameters for other ISMIL algorithms. The fraction of WBCs f was still set to 0.5. COREL_{10} and COREL_{20} were used to represent the datasets for the 10-category and 20-category categorizations, respectively. We found that $C = 2^4$ and $\gamma = 2^{-6}$ gave the minimum twofold cross-validation error on COREL_{10}, and $C = 2^6$ and $\gamma = 2^{-8}$ on COREL_{20}. All these parameter values were fixed in the subsequent experiments.

The average classification accuracies over five different random test sets and the corresponding 95% confidence intervals are provided in Table 3. Overall, the performance of MISUP is competitive with that of other ISMIL algorithms. Specifically, MILIS and MILES outperform MISUP on the COREL_{10} dataset in terms of classification accuracy, meanwhile, MISUP is superior to DD-SVM and MILD. On the COREL_{20} dataset, the performance of MISUP is only worse than that of MILES and better than that of other algorithms. However, the difference between MISUP and MILES is not statistically significant since the 95% confidence intervals for them overlap.

5 Conclusions

In this paper, we have proposed a novel ISMIL algorithm for MIL, MISUP, which is inspired by the self-regulation and suppression mechanisms found in the biological immune system. MISUP is derived by adapting an immune-inspired suppressive algorithm for supervised learning to the MIL setting. The better performance of MISUP for the tasks of drug activity prediction and region-based image categorization demonstrates that our MISUP algorithm is highly comparable with other ISMIL ones in terms of classification accuracy. With

respect to computational efficiency, MISUP is significantly superior to other algorithms based on the better empirical results on the MUSK datasets.

References

1. Dietterich, T., Lathrop, R., Lozano-Pérez, T.: Solving the Multiple Instance Problem with Axis-Parallel Rectangles. Artif. Intell. 89(1-2), 31–71 (1997)
2. Maron, O., Lozano-Pérez, T.: A Framework for Multiple-Instance Learning. In: NIPS, pp. 570–576. MIT Press (1998)
3. Raykar, V., Krishnapuram, B., Bi, J., Dundar, M., Rao, R.: Bayesian Multiple Instance Learning: Automatic Feature Selection and Inductive Transfer. In: ICML, pp. 808–815. Morgan Kaufmann (2008)
4. Rahmani, R., Goldman, S., Zhang, H., Krettek, J., Fritts, J.: Localized Content-Based Image Retrieval. IEEE Trans. Pattern Anal. Mach. Intell. 30(11), 1902–1912 (2008)
5. Viola, P., Platt, J., Zhang, C.: Multiple Instance Boosting for Object Detection. In: NIPS, pp. 1417–1424. MIT Press (2006)
6. Babenko, B., Yang, M., Belongie, S.: Robust Object Tracking with Online Multiple Instance Learning. IEEE Trans. Pattern Anal. Mach. Intell. 33(8), 1619–1632 (2011)
7. Chen, Y., Wang, J.: Image Categorization by Learning and Reasoning with Regions. J. Mach. Learn. Res. 5, 913–939 (2004)
8. Chen, Y., Bi, J., Wang, J.: MILES: Multiple-Instance Learning via Embedded Instance Selection. IEEE Trans. Pattern Anal. Mach. Intell. 28(12), 1931–1947 (2006)
9. Li, W., Yeung, D.: MILD: Multiple-Instance Learning via Disambiguation. IEEE Trans. Knowl. Data Eng. 22(1), 76–89 (2010)
10. Fu, Z., Robles-Kelly, A., Zhou, J.: MILIS: Multiple Instance Learning with Instance Selection. IEEE Trans. Pattern Anal. Mach. Intell. 33(5), 958–977 (2011)
11. Zhang, Q., Goldman, S.: EM-DD: An Improved Multiple-Instance Learning Technique. In: NIPS, pp. 1073–1080. MIT Press (2001)
12. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support Vector Machines for Multiple-Instance Learning. In: NIPS, pp. 561–568. MIT Press (2003)
13. Timmis, J., Hone, A., Stibor, T., Clark, E.: Theoretical Advances in Artificial Immune Systems. Theor. Comput. Sci. 403, 11–32 (2008)
14. Germain, R.N.: An Innately Interesting Decade of Research in Immunology. Nat. Med. 10(12), 1307–1320 (2004)
15. Janeway Jr., C.A., Medzhitov, R.: Innate Immune Recognition. Annu. Rev. Immunol. 20, 197–216 (2002)
16. Janeway, C.A., Travers, P., Walport, M., Shlomchik, M.J.: Immunobiology: The Immune System in Health and Disease. Garland Science (2005)
17. Figueredo, G.P., Ebecken, N.F., Augusto, D.A., Barbosa, H.J.: An Immune-Inspired Instance Selection Mechanism for Supervised Classification. Memetic Computing 4(2), 135–147 (2012)
18. Wang, J., Zucker, J.: Solving the Multiple-Instance Problem: A Lazy Learning Approach. In: ICML, pp. 1119–1126. Morgan Kaufmann (2000)
19. Blake, C., Merz, C.: UCI Repository of Machine Learning Databases (1998), `http://www.ics.uci.edu/~mlearn/`
20. Chang, C., Lin, C.: LIBSVM: A Library for Support Vector Machines. Software (2012), `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Contextual Bandits for Context-Based Information Retrieval

Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gançarski

Department of Computer Science, Télécom SudParis, UMR CNRS Samovar,
91011 Evry Cedex, France
{Djallel.Bouneffouf,Amel.Bouzeghoub,
Alda.Gancarski}@it-sudparis.eu

Abstract. Recently, researchers have started to model interactions between users and search engines as an online learning ranking. Such systems obtain feedback only on the few top-ranked documents results. To obtain feedbacks on other documents, the system has to explore the non-top-ranked documents that could lead to a better solution. However, the system also needs to ensure that the quality of result lists is high by exploiting what is already known. Clearly, this results in an exploration/exploitation dilemma. We introduce in this paper an algorithm that tackles this dilemma in Context-Based Information Retrieval (CBIR) area. It is based on dynamic exploration/exploitation and can adaptively balance the two aspects by deciding which user's situation is most relevant for exploration or exploitation. Within a deliberately designed online framework we conduct evaluations with mobile users. The experimental results demonstrate that our algorithm outperforms surveyed algorithms.

Keywords: Information retrieval, machine learning, exploration/exploitation dilemma, artificial intelligence, reinforcement learning.

1 Introduction

Research at the intersection of information retrieval (IR) and Multi-Armed Bandit problem (MAB) has increasingly engaged the interest of scientists. Authors in [3] consider the IR problem of online learning to rank. For the research on contextual bandits, this work has opened a realistic application area. Users submit queries to an IR system, which construct a documents list ranked according to the query. Then, the interactions of the user with this list can be used to infer feedbacks about the ranking. These feedbacks are then applied to learn better rankings.

The main new challenge for the existent contextual bandit algorithms is to construct result lists from several documents, so that one result list contains both exploratory and exploitative documents and the algorithms have to choose the number of each of them in that list. In this setting, we propose to study this problem in Context-Based Information Retrieval (CBIR) which, to the best of our knowledge, is not yet addressed. We introduce in this paper an algorithm named CBIR-ε-greedy that achieves this goal by balancing adaptively the ranking of exploration/exploitation

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 35–42, 2013.

(exp/exp) of documents according to the user's situation. This algorithm adapts the ε-greedy to CBIR area by selecting suitable user's situations for either exploration or exploitation.

The rest of the paper is organized as follows. Section 2 gives the key notions used throughout this paper. Section 3 reviews some related works. Section 4 presents our CBIR model and describes the algorithms involved in the proposed approach. The experimental evaluation is illustrated in Section 5. The last section concludes the paper and points out possible directions for future work.

2 Key Notions

In this section, we briefly sketch the key notions that will be of use in this paper.

The user's model is structured as a case base, which is composed of a set of situations with their corresponding user's interests, denoted $U = \{(S^i; UI^i)\}$, where S^i is the user's situation and UI^i its corresponding user's interests.

The user's interests are represented using the most representative terms derived from the assumed relevant documents in a particular search situation. In particular, let q^i be the query submitted by a specific user to the retrieval situation S^i. We assume that a document retrieved by the search engine with respect to q^i is relevant if it causes a user's click. Let D^i be the set of assumed relevant documents in situation S^i. Then, UI^i (the user's interests) corresponds to the vector of weighted terms in D^i, where the weight w_{tm} of term tm is computed as follows:

$$w_{tm} = \frac{1}{|D^i|} \sum_{d \in D^i} tf(tm, d) * \log(n / n_{tm}) \quad (1)$$

In Eq.1, $tf(tm; d)$ is the frequency of term tm in document $d \in D^i$, n is the number of documents in the collection, n_{tm} is the number of documents in the collection containing tm. According to Eq. 1, each document $d \in D^i$ is represented by a term vector where the relevance value of each term tm in situation S^i is computed using the $tf*idf$ weighting.

The user's context has a multi-ontology representation where each ontology corresponds to a context dimension: $C=(\boldsymbol{O}_{Location}, \boldsymbol{O}_{Time}, \boldsymbol{O}_{Social})$. Each dimension models and manages a different context information type, namely location, time, and social information. We focus on these three dimensions since they cover all the needed information for our application domain (detailed in Section 5).

The user's situation is an instantiation of the user's context. We consider a situation as a triple $S = (\boldsymbol{O}_{Location}.x_i, \boldsymbol{O}_{Time}.x_j, \boldsymbol{O}_{Social}.x_k)$ where x_i, x_j and x_k are ontology concepts or instances. As an example, suppose the following data are sensed from the user's mobile phone: the GPS shows the latitude and longitude of a point "48.89, 2.23"; the local time is "Oct_3_12:10_2012" and the calendar states "meeting with Paul Gerard". The corresponding situation is:

$S = (\text{"48.89,2.23"}, \text{"Oct\_3\_12:10\_2012"}, \text{"Paul\_Gerard"})$. To build a more abstracted situation, we interpret the user's behavior from this low-level multimodal

sensor data using ontology reasoning means. For example, from *S*, we obtain the following situation: *Meeting = (Restaurant, Work_day, Financial_client)*.

Among the set of captured situations, some of them are characterized as High-Level Critical Situations.

High-Level Critical Situations (HLCS) are situations where the user needs the most suitable information delivered by the IR system. In such situations (e.g. a professional meeting), the system must exclusively perform exploitation rather than exploration-oriented learning. In the other case, where the user is for instance using his/her information system at home, on vacation with friends, the system can make some exploration by retrieving some information ignoring his/her interests. HLCS are predefined by the domain expert. In our case, we conduct a study with professional mobile users, which is described in detail in Section 5. As examples of HLCS, we can find *S1 = (restaurant, midday, client)* or *S2 = (company, morning, manager)*.

3 Related Work

We refer, in the following, an overview of the existing bandit algorithms and recent techniques that consider the user's context for ranking result in IR.

Bandit Algorithms Overview. The MAB problem was originally described by Robbins [8]. The *ε-greedy* is one of the most used algorithms to solve the bandit problem and was first described in [9]. The *ε-greedy* strategy chooses a random document with epsilon-frequency (ε), and chooses the document with the highest estimated mean otherwise. The estimation is based on the rewards observed thus far. ε must be in the interval [0, 1] and its choice is left to the user. Authors in [10] extend the ε-greedy strategy by dynamically updating the ε exploration value. At each iteration, they run a sampling procedure to select a new ε from a finite set of candidates. The probabilities associated to the candidates are uniformly initialized and updated with the Exponentiated Gradient (EG) [5]. This updating rule increases the probability of a candidate ε if it leads to a user's click.

Compared to the standard multi-armed bandit problem, the CBIR does not select individual documents, but constructs result lists from several documents, so that one result list contains both exploratory and exploitative documents. Therefore, the bandit algorithms need to be modified to manage this new challenge. As far as we know, no existing works address the problem of exr/exp tradeoff in CBIR, except a recent research work that studied the contextual bandit problem in IR area. Indeed, authors in [3] have proposed to adapt the ε-greedy approach to their need. They maintain two document lists, one exploitative (based on the currently learned best ranking), and one exploratory (introducing variations to the current best ranking to explore potential improvements). An exploration rate ε determines the relative number of documents each list contributes to the final list shown to the user. However this rate is just left to the user and is not deeply studied.

Result Ranking in CBIR. The need of context in IR is ineluctably increased due to the wide manipulation of smartphones. Different recent works address the challenge of ranking IR results according to the user's context.

In [7], a linear function of reordering is adopted to adjust the search results to the user. The initial score of the document is multiplied by a score of customization, which is a linear combination as a weighted sum of the preferences of user content and location. To rank documents in [7], a pre rank is calculated to measure how much a document is related to a particular context and the post rank is measured to know how much a document is related to user query. The total rank weight is calculated by summing up pre rank weight and post rank weight. Documents are ranked in descending order of their final rank weight.

In [2] the contextual user's profile is used to reorder the search results. They propose a linear combination of the original result returned by a conventional IR with a score of customization calculated between the document and the profile of the user's document ranking. The personalization score is calculated for each document with the contextual user's profile based on a similarity measure between the vector of weighted concepts of the document and the user profile is also represented as a vector of weighted concepts.

As shown above, none of the mentioned works address the exr/exp problem in CBIR. This is precisely what we intend to do in our approach: we propose to consider the criticality of the user's situation when managing the exr/exp-tradeoff to rank the results. This strategy achieves high exploration when the current user's situation is not critical and achieves high exploitation in the inverse case.

4 CBIR Model

In our CBIR, the selection and ranking of documents is modeled as a contextual bandit problem including user's situation information. Formally, a bandit algorithm proceeds in discrete trials $t = 1 \ldots T$. For each trial t, the algorithm performs the following tasks:

Task 1: Let S^t be the current user's situation when he/she submits a request, and PS the set of past situations. The system compares S^t with the situations in PS in order to choose the most similar one, S^p:

$$S^p = \underset{S^c \in PS}{\arg\max} \left(sim(S^t, S^c) \right) \tag{2}$$

The semantic similarity metric is computed by:

$$sim(S^t, S^c) = \sum_j \alpha_j \cdot sim_j \left(x_j^t, x_j^c \right) \tag{3}$$

In Eq. 3, sim_j is the similarity metric related to dimension j between two concepts x_j^t and x_j^c; α_j is the weight associated to dimension j (during the experimental phase, α_j has a value of 1 for all dimensions). This similarity depends on how closely x_j^c and x_j^c are related in the corresponding ontology. We use the same similarity measure as [1] defined by:

$$sim_j \left(x_j^t, x_j^c \right) = 2 * \frac{deph(LCS)}{(deph(x_j^c) + deph(x_j^t))} \tag{4}$$

In Eq. 4, LCS is the *Least Common Subsumer* of x_j^t and x_j^c, and *deph* is the number of nodes in the path from the node to the ontology root.

Task 2: Let D be the documents collection. After retrieving S^p, the system observes the corresponding user's interests UI^p in the user's model case base. Based on the observed UI^p and the query q, the algorithm ranks documents in D using the traditional cosine similarity measure.

Task 3: From the ranked list of documents presented to the user, the algorithm receives the set D^t of clicked documents and improves its document-selection strategy with the new observation: in situation S^t, documents in D^t obtain a user's click. Depending on the similarity between the current situation S^t and its most similar situation S^p, two scenarios are possible: (1) If $sim(S^t, S^p)<3$: the current situation does not exist in the case base; the system adds this new case composed of the current situation S^t and the current user's interest UI^t computed from the set D^t of clicked documents using Eq. 1; (2) If $sim(S^t, S^p) = 3$: the situation exists in the case base; the system updates the case having as premise the situation S^p with the current user's interest UI^t, the update being done by integrating the new documents, $D^p = D^p \cup D^c$, and computing the new vector UI^t using Eq. 1.

4.1 The IR-ε-Greedy Algorithm

The *IR-ε-greedy* algorithm ranks documents using the following equation:

$$SCORE(d_i) = \begin{cases} result_q(q,d_i) & if\ (l < j; l < \varepsilon) \\ result_c(UI^p, d_i) & if\ (l \geq j; l < \varepsilon) \\ random(d_i) & if\ (l \geq \varepsilon) \end{cases} \quad (5)$$

In Eq. 5, $i \in \{1,\ldots N\}$ where N is the number of documents in the collection, $d_i \in D^t$, l and j are random values uniformly distributed over $[0, 1]$ which define the exr/exp tradeoff; $\varepsilon \in [0, 1]$ is the probability of making a random exploratory rank of documents. $result_q(q,d_i)$ gives the original score returned by the system based on the query q using the cosine similarity as follows:

$$result_q(q,d_i) = \frac{q \cdot d_i}{\|q\| \, \|d_i\|} \quad (6)$$

$result_c(UI^p,d_i)$ gives the contextualization score returned by the system based on the user's interests UI^p, and it is also computed using cosine similarity as follows :

$$result_c(UI^p,d_i) = \frac{UI^P \cdot d_i}{\|UI^p\| \, \|d_i\|} \quad (7)$$

$random(d_i)$ gives a random rank to the document d_i to perform exploration.

4.2 The CBIR-ε-Greedy Algorithm

To improve the adaptation of the *IR-ε-greedy* algorithm to HLCS situations, the *CBIR -ε-greedy* algorithm compares the current user's situation S^t with $S^m \in$ HLCS, which is the centroid of the HLCS situations (the centroid is computed using k-mean algorithm [4]).

Depending on the similarity between the S^t and S^m, two scenarios are possible:

(1) If $sim(S^t, S^m) \geq B$ (B is the similarity threshold), the current situation is critical; the *IR-ε-greedy* algorithm is used with $\varepsilon=0$ (exploitation) and S^t is inserted in the HLCS set of situations.
(2) If $sim(S^t, S^m) < B$, the current situation is not critical; the *IR-ε-greedy* algorithm is used with $\varepsilon>0$ (exploration) computed as indicated in Eq. 8.

$$\varepsilon = \begin{cases} \varepsilon_{\max} - \left(\dfrac{sim(S^t, S^m)}{B} \right) & if\ (sim(S^t, S^m)) < B.\varepsilon_{\max} \\ 0 & otherwise \end{cases} \quad (8)$$

In eq 5, $\varepsilon_{max} \in [0,1]$ is the maximum of allowed exploration. Both ε_{max} and B are computed using EG [5].

To summarize, the system does not make exploration when the current user's situation is critical; otherwise, the system performs exploration. In this case, the degree of exploration decreases when the similarity between S^t and S^m increases.

5 Experimental Evaluation

In order to empirically evaluate the performance of our approach, and in the absence of a standard evaluation framework, we propose an online evaluation framework. The main objective of the experimental evaluation is to evaluate the performance of the proposed algorithm (*CBIR-ε-greedy*). In the following, we present and discuss the obtained results.

We have conducted a diary study with the collaboration of the French software company Nomalys[1]. This company provides an enterprise search engine application that allows users to connect their information system throw mobile devices.

We conduct our experiment with 3500 users of Nomalys. We compare *CBIR-ε-greedy* to a variant *IR-ε-greedy* using existing ε trade-off described in the related work. To this end, we have randomly split users on three groups: the first group has an IR system with the *CBIR-ε-greedy*; the second group is equiped with *IR-ε-greedy*, where ε is computed using EG [5], what we call *EG-IR-ε-greedy*; finally,the last group uses the *IR-ε-greedy* with exploration $\varepsilon=0$ (non exploration algorithm, baseline). Note that we do not evaluate the algorithm proposed in [3] because it does not consider the variance of the ε, wich is our goal in this evaluation.
The experimental evaluation was carried out, as follows.

[1] Nomalys is a company that provides a graphical application on Smartphones allowing users to access their company's data.

Number of Clicks on the Top 10 Documents. We compare the algorithms regarding the number of clicks on the top 10 documents. In Fig. 1, the horizontal axis represents the day of the month and the vertical axis is the performance metric, which is the number of clicks on the top 10 documents per the number of times the users make a request.

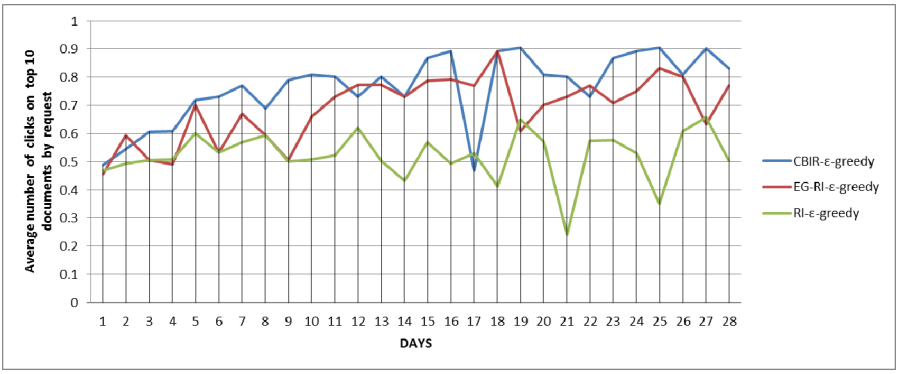


Fig. 1. Average number of clicks on top 10 documents for exr/exp algorithms

We have several observations regarding the different exr/exp algorithms. Overall, tested algorithms have better performances than the baseline. While the *EG-IR-ε-greedy* algorithm converges to a higher number of clicks compared with a baseline, its overall performance is not as good as *CBIR-ε-greedy*. We have also observed the average number of clicks per request for all the 28 days and we observe that the *CBIR-ε-greedy* algorithm effectively has the best average number of clicks during this month, which is 0.77 click/request. *EG-IR-ε-greedy* obtains 0.69 click/request and the baseline, 0.5141 click/request. *CBIR-ε-greedy* increases the average click/request by a factor of 1.5 over the baseline and outperforms *EG-IR-ε-greedy* algorithm. The improvement comes from a dynamic tradeoff between exr/exp on documents ranking, controlled by the critical situation (HLCS) estimation. The algorithm takes full advantage of exploration document in non-critical situations, without wasting opportunities to establish good results in critical situations.

6 Conclusion

In this paper, we studied the problem of exploitation and exploration in Context-Based Information Retrieval and proposed a novel approach that ranks documents by balancing adaptively exr/exp regarding the user's situation. For that, we take into account critical situations, where the user needs to get suitable information and thus exploration should not be performed. We have presented an evaluation protocol based on real mobile user. We evaluated our approach according to the proposed evaluation protocol. This study yields to the conclusion that considering the exploration/exploitation tradeoff, significantly increases the performance of the CBIR.

In the future, we plan to improve the notion of the HLCS in the scope of CBIR by introducing more contextual dimensions.

References

1. Bouneffouf, D., Bouzeghoub, A., Gançarski, A.L.: A Contextual-Bandit Algorithm for Mobile Context-Aware Recommender System. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part III. LNCS, vol. 7665, pp. 324–331. Springer, Heidelberg (2012)
2. Goenka, K., Arpinar, I.B., Nural, M.: Mobile web search personalization using ontological user profile. In: Proceedings of the 48th ACM Southeast Conference (2010)
3. Hofmann, K., Whiteson, S., Maarten, R.: Contextual Bandits for Information Retrieval. In: NIPS 2011 Workshop (2011)
4. Kanungo, T., David, M., Nathan, S., Christine, D.: An Efficient k-Means Clustering Algorithm: Analysis and Implementation. IEEE Trans. Pattern Anal., 881–892 (2002)
5. Kivinen, J., Manfred, K.: Exponentiated gradient versus gradient descent for linear predictors. Information and Computation 132 (1995)
6. Leung, K., Lee, D., Lee, W.: Personalized web search with location preferences. In: Proceedings of the 26th International Conference on Data Engineering (ICDE), pp. 701–712. IEEE (2010)
7. Rani, S., Jain, V., Gandhi, G.: Context Based Indexing and Ranking in Information Retrieval Systems. International Journal of Computer Science and Management Research 2(4) (April 2013)
8. Robbins, H.: Some aspects of the sequential design of experiments. Bulletin of the American Mathematical Society 58, 527–535 (1952)
9. Watkins, C.: Learning from Rewards. Ph.D. thesis. Cambridge University (1989)
10. Wei, L., Wang, X., Zhang, R., Cui, Y., Mao, J., Jin, R.: Exploitation and Exploration in a Performance based Contextual Advertising System. In: Proceedings of the International Conference on Knowledge Discovery and Data Mining, pp. 27–36. ACM, New York (2010)

Density Estimation Method Based on Self-organizing Incremental Neural Networks and Error Estimation

Xiong Xiao, Hongwei Zhang, and Osamu Hasegawa

Imaging Science and Engineering Laboratory, Tokyo Institute of Technology, Japan
xiaoxiong_sysu@yahoo.co.jp, {zhang.h.ac,O.Hasegawa}@m.titech.ac.jp
http://haselab.info/

Abstract. In this paper, we propose an analysis of a self-organizing incremental neural network(SOINN), using new network adjusting algorithms, and a batched density estimation method combined with kernel density estimation(KDE) to simulate the performance of an SOINN. To evaluate the density estimation method, a quantitative relationship between absolute error and the number of neurons in the network is given which shows that any accuracy can be satisfied by adjusting networks. Although the precise result relies on information of the input distribution, the analysis method can be applied to any continuous input situation. After the error estimation, a comparison between the two new network adjusting algorithms is provided, and experiment results are provided and discussed.

Keywords: Self-organizing incremental neural network(SOINN), Density esimation, Unsupervised learning.

1 Introduction

A self-organizing incremental neural network(SOINN) is an unsupervised learning algorithm originally proposed by Shen and Hasegawa[1] based on a self-organizing map[2] and the growing neural gas concept[3]. The advantages of an SOINN are that it does not require a pre-determined clusters number k, it is suitable for online learning tasks, and it has high noise tolerance.

The original SOINN consisted of a two-layer structure and succeeded in obtaining an approriate cluster number without the need to predetermine the structure of the network. An enhaced-SOINN[4] simplifies an SOINN to a one-layer structure with less parameters, and can distinguish between high-density overlapped clusters. Later, a more simplified algorithm, the Adjusted SOINN Classifier[5] with only two parameters has been proposed and has become the fundamental algorithm for continued studies on an SOINN. Sudo, Sato and Hasegawa have proposed a method to combine associative memory with an SOINN[6] that has led to an application in robot navigation[7].

However, to the best of our knowledge, there has been no mathematical analysis of an SOINN; thus, its overall performance remains unclear. This paper

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 43–50, 2013.

presents an analysis of an SOINN to fill in the blanks, and we present a density estimation algorithm to evaluate the performance of an SOINN.

2 SOINN Algorithm

This section presents the most common one-layer algorithm, or the fundamental SOINN algorithm.

***Algorithm* 2.1** Fundamental Algorithm

(1) ***Parameters*** : A: set of all neurons. $C \subset A \times A$: set of all edges. N_i: set of all neighbors of neuron i. W_i: weight of neuron i. λ: time period to delete redundant neurons. age_{max}: parameter to delete edges.
(2) if first time of input then
(3) $A \leftarrow c_1, c_2$; randomly pick up two vectors from trainning data to initialize the neuron set.
(4) $C \leftarrow \emptyset$
(5) end if
(6) while input data ξ exist do
(7) $s_1 \leftarrow arg\min_{c \in A} ||\xi - W_c||$: find out the winner.
(8) $s_2 \leftarrow arg\min_{c \in A \setminus s_1} ||\xi - W_c||$: find out the second winner.
(9) calculate similarity thresholds T_{s_1}, T_{s_2}. If i got neighbors, T_i is the distance to the farest neighbor, else the distance to the nearest neuron.
(10) if $||\xi - W_{s_1}|| > T_{s_1}$ or $||\xi - W_{s_2}|| > T_{s_2}$ then
(11) $A \leftarrow A \cup \xi$: insert ξ as a new neuron.
(12) else
(13) if $(s_1, s_2) \notin C$: there is no edge between the winner and second winner, then
(14) $C \leftarrow C \cup (s_1, s_2)$: add new edge into the network
(15) end if
(16) $age_{(s_1,s_2)} \leftarrow 0$: reset the age of (s_1, s_2)
(17) $age_{(s_1,i)} \leftarrow age_{(s_1,i)} + 1(\forall i \in N_{s_i})$: increase age of edges connected with the winner by 1.
(18) $\triangle W_{s_i} = \epsilon(t_{s_1})(\xi - W_{s_1}), \triangle W_i = \epsilon(100t_i)(\xi - W_i)(\forall i \in N_{s_1}), \epsilon(t) = \frac{1}{t}$
(19) using $vartriangle W_{s_i}, \triangle W_i$ to adjust the winner and it's neighbors
(20) delete edge whos age is larger than age_{max}
(21) among these neurons which the edge deleted in last step connected to, delete neurons having no neighbors.
(22) if input data number becomes $n \times \lambda (n \in N^+)$ then
(23) delete neurons having less than one neighbor
(24) end if
(25) end while

2.1 Two New Network Control Algorithm

In the standard algorithm, the Voronoi region of each neuron is highly dependent on the input sequence of data. Therefore, two new network control algorithms are proposed to adjust the network in its topology and to improve its statistics.

One algorithm is an edge-length control method. During every time period when the SOINN begins deleting redundant neurons, this algorithm will check the length of every edge, and if one edge is n times longer than the average edge-length, a new neuron, with a learning time based on the two end neurons, will be inserted to divide the edge, and the learning time of the the neurons will be reduced. Using this algorithm, distance between neurons become similar.

Another algorithm is a learning time control method. It check learning time instead of edge length to insert new neurons. Moreover it only inserts one new neuron each time, it is invoked to reduce the influence of low probability situations. Using this algorithm, the probabilities of the Voronoi regions become similar.

2.2 Analysis of the SOINN Algorithm

In the SOINN algorithm, the movement of neurons uses the following formulars:

$$\triangle W_c = \epsilon(t)(\xi_t^c - W_c) \tag{1}$$

$$W_c(0) = (random\, signal\, according\, to\, p(\xi)) \tag{2}$$

$$W_c(1) = W_c(0) + \epsilon(1)(\xi_1^c - w_c(0)) \tag{3}$$

$$= \xi_i^c \tag{4}$$

$$W_c(2) = W_c(1) + \epsilon(2)(\xi_2^c - W_c(1)) \tag{5}$$

$$= \frac{\xi_1^c + \xi_2^c}{2} \tag{6}$$

$$\vdots \tag{7}$$

$$W_c(t) = \frac{\xi_1^c + \xi_2^c + \cdots + \xi_t^c}{t} \tag{8}$$

It shold be noted that the set of signal $\xi_1^c, \xi_2^c, \ldots, \xi_t^c$ for which a neuron c has been chosen as the winner may contain elements that lie outside the current Voronoi region of c. The reason is that each movement of W_c changes the borders of the Voronoi region V_c.Therefore, although $W_c(t)$ represents the mean of the signals when a neuron has been a winner, it does not represent the arithmetic mean of the neuron's current Voronoi region V_c.

Another impotant point about the movement formulars is that there is no strict convergence, because of the divergence:

$$\lim_{n\to\infty} \sum_{t=1}^{n} \frac{1}{t} = \infty. \tag{9}$$

However, in simulatons where the signal distribution is stationary, it has been shown that k-means, using the same moving formulars (1)-(8), asymptotically converges to a configuration where each reference vector W_c is positioned such that it coincides with the expectation value(10) of its Voronoi region V_c[8]. Because the SOINN also adapts nodes, using a learning rate of $\frac{1}{100t}$, when their neighbors become winner, this expectation value can be formulated as (11)-(13)

$$E(\xi|\xi \in V_c') = \int_{V_c'} \xi p'(\xi) d\xi \tag{10}$$

$$V_c' = V_c \cup \bigcup_{i \in Neighbor_c} V_i \tag{11}$$

$$p'(\xi) = \begin{cases} \frac{1}{C} p(\xi), & \xi \in V_c \\ \\ \frac{1}{C} \frac{p(\xi)}{100}, & \xi \in V_i, i \in Neighbor_c \end{cases} \tag{12}$$

$$C = P(V_c) + \sum_{i \in Neighbor_c} \frac{P(V_i)}{100}. \tag{13}$$

In addition, as neurons are generated and ejected during the learning process, the use of learning time to represent the probability of each Voronoi region causes some unpredictable errors. Therefore, a method is proposed to overcome these problems.

3 Density Estimation

Although the SOINN is not strictly proven in mathematics, it can infer the region and density of the input data distribution very rapidly while ignoring external noise, and neurons can be controlled during learning process to a certain degree. Therefore, this method divides the density estimation process into two parts. First, input a percentage of data set to obtain neurons for the SOINN, and use the remaining data to calculate Gaussian kernels. Figure.2 shows some experiment results of this method.

***Algorithm* 3.1.** Gaussian kernel calculation algorithm

(1) Obtain the network information from the SOINN, calculate the threshold region of each neuron(,***Algorithm* 2.1**), create a shadow upon each neuron that has the same wheight and initialized learning time as its parent neuron for shadow movments, and collect height infomation h for region probability estimation.
(2) Count the number n of threshold regions present in the current input, and increase learning time and h of these neurons by 1 and $\frac{1}{n}$, respectively, all of these neurons become winners.
(3) Move the shadows of every winner as specified by formula (1); these shadows will represent the arithmetic mean of each threshold region.
(4) Each neuron will record input data within its threshold region.

(5) Finally, when the input concludes, use the weights of shadows and the input records of the neurons to calculate variance and covariance of each Gaussian kernel. Density estimation $p^*(x)$ is the summation of these kernels.

4 Error Estimation

This section presents a relationship between the abusolute error of this density estimaiton method and the number of neurons, considering input data as a 2D standard normal distribution. Because the precise topological information of a neural network is unpredictable and relies on the input sequence, the error simulation depens on some assumptions:

(1) Assume that the edge-length control method is used in the SOINN, and creats edges of exact the same length. Thus the input distribution region is divided into regular equilateral triangles.
(2) Assume each Voronoi region has uniform distribution. The reason is that the original Normal distribution will require the solution of many transcendental equations otherwise.

Under these assumptions, the error estimaiton will never be precise; thus, an upper bound must be defined.

Under assumption(1), the input region can be considered to be divided by three sets of parallel lines; and lines belonging to two different sets are separated by an angle of 60°. Each Voronoi region is thus a regular hexagon, and each threshold region is a circle with a radius equal to the edge length.

Under assumption(2), the expectation point of each Gaussian kernel will be the weight of neuorn, and the variance can be calculated as:

$$\delta^2 = \int_{T_c} \frac{1}{\pi r^2} \xi^2 d\xi = \frac{r}{\sqrt{3}} \tag{14}$$

where r is the length of every edge when all edges have equal length, Threshold area T_c is a circle with a radius of r.

The value h of each neuron is similar to the probability of each Voronoi region, and it is easy to prove that if both the number of input data and neurons is infinity, h and the probability are equal.

Now, with the assumptions discussed above, consider each Voronoi region. The probability of the summation of Gaussian kernels is $Ch_c + \frac{1-C}{6} \sum_{i \in N_c} h_i$, thus the absolute error is

$$Error(V_c) = \left| h_c - \frac{C'}{6} \sum_{i \in N_c} h_i \right| \tag{15}$$

$$\leqslant \frac{C'}{6} \sum_{i \in N_c} |h_c - h_i| \tag{16}$$

$$C' = 1 - \int_{V_c} p_{Gaussian\,kernel}(\xi) d\xi \tag{17}$$

Futhermore, each neuron has at most three neighbors with larger values of h, and

$$\frac{\partial^2 Normal(x,y)}{\partial x\,\partial y} \leqslant \frac{d\,\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}}{dx} = -\frac{x}{\sqrt{2\pi}}e^{-\frac{x^2}{2}} \tag{18}$$

$$\leqslant -\frac{1}{\sqrt{2\pi}}e^{-\frac{1^2}{2}} \leqslant \sqrt{3}-1, \tag{19}$$

The error can be more strictly defined by

$$Error(V_c) \leqslant \frac{C'}{6\times 2}\sum_{i\in N_c}|h_c - h_i| \tag{20}$$

Consider each line in these line sets under assumption(1). There is(are) one or two maximum h value neuron(s). Thus,

$$\sum_{i\in L_c}\sum_{j\in L_c\cap N_i}|h_i-h_j| = \begin{cases} 4h_c, & h_c = \max_{i\in L_c} hi \\ 2(h_c+h_c'), & h_c = h_c' = \max_{i\in L_c} hi \end{cases} \tag{21}$$

Combining the three line sets:

$$Error \leqslant \frac{C'}{12}\sum_{i\in N_c}|h_c-h_i| \tag{22}$$

$$\leqslant 3\times\frac{C'}{12}\sum_{L_c\in Line\ set}\sum_{i\in L_c}\sum_{j\in L_c\cap N_i}|h_i-h_j| \tag{23}$$

$$\leqslant 3\times 4\times\frac{C'}{12}\int_{-\frac{r}{2}}^{\frac{r}{2}}\int_{-\infty}^{\infty}Normal(x,y)dxdy \tag{24}$$

$$= C'\cdot erf(\frac{r}{2\sqrt{2}\delta_{true}}) \to 0(\text{Number of neurons}\to\infty, r\to 0) \tag{25}$$

$$C' = 1-\int_{V_c}p_{Gaussian\ kernel}(\xi)d\xi \approx 1-0.3127 \tag{26}$$

Although the last result is related to input distribution, in this paper, a standard normal distribution, the inferring method can be applied to continuous input situations. In fact, continuous distribution can be considered as a combination of sets of normal distributions. In addition, if using the learning time control method, the area or probability of the Voronoi regions, where the input density is higher, will be smaller; therefore, performance will be improved.

4.1 Experimental Results

In this experiment, the input vector is generated from a 2D standard normal distribution, and the absolute error is calculated using $\int|N(\xi)-p^*(\xi)d\xi|$, $p^*(\xi)$ is the estimation result. Input number is 100,000. The number of neurons under the same parameter sets varies in a short range.

In Figure.1 the x-value of each error bar is the average number of neurons under one parameter set. Figure.2 shows some experimental results of density estimation.

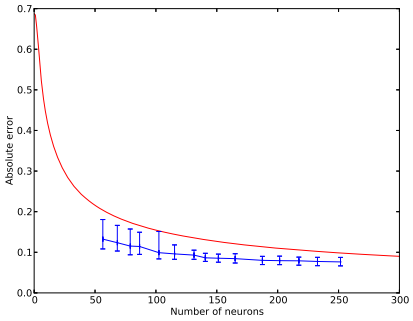
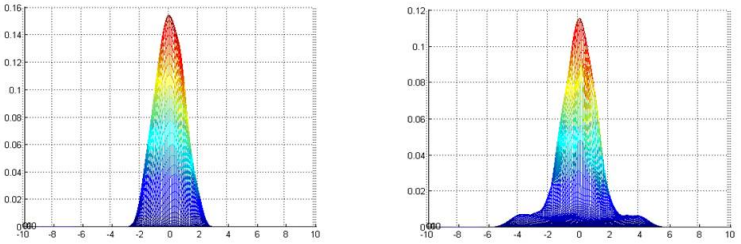


Fig. 1. Relationship between absolute error and number of neurons. The upper curve is the result of proposed error estimation, and the error bars are experimental results with y values of average error, maximum error and minimum error.



(a) (b)

Fig. 2. Experimental results of density estimation:, input data distribution is: (a)standard normal distribution, (b)70% normal distribution + 30% uniform distribution ranged in $[-5, 5] \times [-5, 5]$

In this experiment, using $100,000$ input data, with 100 neurons, the running time was approximately 10 min, using Matlab on a Lenovo y470 laptop, equipped with an Intel i5-2450M 2.5GHz×2 processor and 8G memory. Surprisingly, the same simulation takes less than 10s to complete using C++ code.

5 Conclusion

This paper presents an analysis of the SOINN algorithm, and a two-phase batched density estimation method is proposed to provide an estimation of the quantitative relationship between absolute error and number of neurons. Under the error estimation, with enough input data, any input data distribution can be estimated with an absolute error less than a value soley ralated to the number of neurons. This result can be regarded as a confidence analysis of SOINN.

6 Future Work

This paper is proposed under the assumption that the input data is from a 2D distribution. A high-dimension problem is of future interest. Because an SOINN uses the Euclidean distance, the concentration phenomenon becomes a problem when considering high dimension problems. In addition, in the error estimation section, the first assumption is based on the edge-length control method, perhaps yielding a topology that is too ideal for real world problems. To provide a more rigorous or strict analysis, we plan to evaluate how to realistically represent neurons in networks in the future.

Acknowledgments. This study was sponsored by the Japan Science and Technology Agencys CREST project.

References

1. Shen, F., Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning. Neural Networks 19, 90–106 (2006)
2. Kohonen, T.: Self-organized formation of topologically conrrect feature maps. Biological Cybernetics 43, 59–69 (1982)
3. Frizke, B.: A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems, pp. 625–632 (1995)
4. Shen, F., Ogura, T., Hasegawa, O.: An enhanced self-organizing incremental neural network for online unsupervised learning. Neural Networks 20(8), 893–903 (2007)
5. Shen, F., Hasegawa, O.: A Fast Nearest Neighbor Classifier Based on Self-organizing Incremental Neural Network. Neural Networks 21(10), 1537–1547 (2008)
6. Sudo, A., Sato, A., Hasegawa, O.: Associative Memory for Online Learning in Noisy Environments Using Self-organizing Incremental Neural Network. IEEE Trans. on Neural Networks 20(6), 964–972 (2009)
7. Tangruamsub, S., Kawewong, A., Tsuboyama, M., Hasegawa, O.: Self-Organizing Incremental Associative Memory-Based Robot Navigation. IEICE Trans. on Information and Systems E95-D(10), 2415–2425 (2012)
8. MacQueen: On convergence of k-means and partitions with minimum average variance. Ann. Math. Statist. 36, 1084 (1965)

Genetic Network Programming with Simplified Genetic Operators

Xianneng Li, Wen He, and Kotaro Hirasawa

Graduate School of Information, Production and Systems, Waseda University
{sennou@asagi.,hewen@toki.,hirasawa@}waseda.jp

Abstract. Recently, a novel type of evolutionary algorithms (EAs), called Genetic Network Programming (GNP), has been proposed. Inspired by the complex human brain structures, GNP develops a distinguished directed graph structure for its individual representations, consequently showing an excellent expressive ability for modelling a range of complex problems. This paper is dedicated to reveal GNP's unique features. Accordingly, simplified genetic operators are proposed to highlight such features of GNP, reduce its computational effort and provide better results. Experimental results are presented to confirm its effectiveness over original GNP and several state-of-the-art algorithms.

Keywords: evolutionary algorithms, genetic network programming, directed graph, transition by necessity, invalid evolution.

1 Introduction

As one of the most widely studied fields for solving optimization problems, a large number of evolutionary algorithms (EAs) have been developed. Genetic Algorithm (GA) and Genetic Programming (GP) are the typical examples, which use bit-string and tree structures to represent their individuals. Different EAs would have their own suitable problems, however which cannot guarantee that they are superior in all problems with the No-Free-Lunch Theorem [1].

In recent years, there has been a particular interest to expand EAs by developing more complex structures. One of the directions is to study graph structures [2, 3]. Among different variants, a novel graph-based EA, named Genetic Network Programming (GNP) [3–5], has been proposed recently. GNP develops a distinguished directed graph, which is composed of two kinds of nodes: judgment and processing node, consequently allowing the flexible representation and recombination of "if-then" type decision rules. The separation of judgment and processing provides the fundamental basis of GNP to efficiently evolve the compact programs by only transiting the necessary judgments and processing. It was first developed to solve the problems of controlling the agents' behavior [4, 5], while in recent years it has been extended to many problems [6–9].

Although GNP has been studied extensively, one lack of its research is to reveal the fundamental features of GNP based on its distinguished directed graph. Particularly its unique feature of "transition by necessity" is seldom addressed,

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 51–58, 2013.

which inspires the work of this paper. Moreover, all the current works of GNP apply traditional genetic operators, i.e., uniform crossover/mutation, to evolve the directed graphs, regardless its unique features. This paper provides a discussion of GNP, and proposes simplified genetic operators by taking its distinguished features into account. By explicitly addressing its unique features, the proposed methods allow GNP to have better evolution efficiency for finding better results.

GNP and its evolution are briefly revisited in section 2. Section 3 discusses the drawbacks of original GNP and proposes the simplified genetic operators. Section 4 evaluates this work in a benchmark testbed – Tileworld [4, 5, 10] – of GNP. Finally the conclusions and potential future work are presented.

2 Genetic Network Programming (GNP) Revisited

2.1 Directed Graph Structure

Let G represent the directed graph structure of GNP:

Definition 1. *G is defined by a tuple $(N_{\text{node}}, LIBRARY)$, where N_{node} represents the set of nodes in G. LIBRARY denotes functions given by the problems.*

Definition 2. *Each node i is defined by $(NT_i, NF_i, B(i), C_i)$. $NT_i \in \{1, 2\}$ defines the node type for judgment/processing. NF_i denotes its function. $B(i)$ represents the set of branches in node i. C_i represents the connection of node i.*

Definition 3. *The connection information C_i of node i is a set of C_{ik} indicating the node connected from the k_{th} branch of node i. It is noted that $|C_i| = |B(i)|$.*

In GNP, all terms except C, that is, a set of C_i, are predefined. In other words, the node connections are evolved by changing the variable C of each node.

Definition 4. *The **search dimensions** of GNP is represented by the number of branches in G, i.e., $\sum_{i \in N_{\text{node}}} |B(i)|$.*

Such a distinguished directed graph consists of several unique features. First, GNP can avoid the bloat problem of GP even with the fixed size of G [3]. Second, it is capable of efficiently realizing the repetitive processes based on the frequent reuse of nodes. Most importantly, empirical results have confirmed that evolving the node connections allows GNP to efficiently generate the compact programs for problem solving. Fig. 1 presents an example of the directed graph G of GNP.

2.2 Transition by Necessity

Although GNP has been studied extensively in both its enhancements and applications, one lack of its current research is to intensively reveal the fundamental features of GNP from the perspective of its distinguished graph structure.

Applying GNP for problem solving is derived by following a sequence of node transitions in each individual. In other words, the core of each directed graph

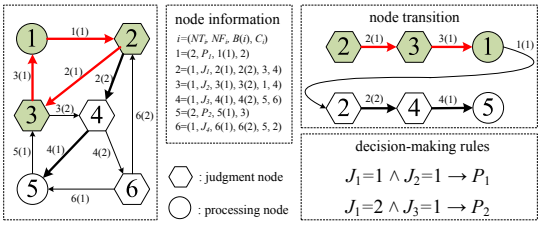


Fig. 1. GNP architecture

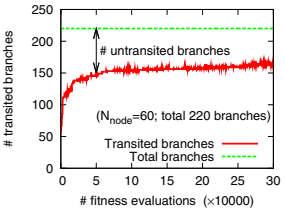


Fig. 2. "*Transition by necessity*"

is its node transitions obtained. By the separation of judgments and processing, the node transitions are equivalent to a sequence of "if-then" type decision rules. More importantly, for one particular task, only a part of the nodes and its transitions are activated. This is the most important feature of GNP, called "**transition by necessity**", inspired by brain science that human brain works in a way of functions distribution, where only specific parts of neurons are activated corresponding to certain sorts of sensory information the brain perceives [11].

Fig. 1 shows an example of GNP, where only parts of structures are transited, and some nodes are frequently reused (e.g., node 2) while the activated rules explicitly shows more *generalized* forms than the traditional methods taking all judgments into account. Fig. 2 presents an empirical example in Tileworld. Consequently, the directed graph of GNP shows three unique features: 1) *functions distribution*, 2) *efficient reusability of nodes* and 3) *generalization ability*.

2.3 Evolution of Original GNP

As the other EAs, GNP applies the nature of evolution to iteratively evolve the population *Pop* from scratch. In every generation, *Pop* is subject to evolution by two genetic operators, i.e., crossover and mutation. Though various kinds of genetic operators can be applied to evolve GNP, it has been confirmed that uniform genetic operators are capable of obtaining better results than the others [12], which soon become the standard of GNP for problem solving [4, 5].

3 Simplified Genetic Operators

It is notable that the traditional genetic operators does not take the unique features of GNP into account, which brings the unnecessary difficulty for evolution.

The most critical drawback of uniform genetic operators in GNP is the one we define as *invalid evolution*. For any individual g, let TB_g and UB_g denote the sets transited and untransited branches in a given task, where these two disjoint sets are the partition of the search dimensions $\sum_{i \in N_{\text{node}}} |B(i)|$.

Clearly, traditional genetic operators treat all branches equally to evolve g, regardless the fact that the fitness of g in a given task is only based on the

evaluation of its node transitions, i.e., TB_g. In other words, UB_g has equal opportunity to evolve, even they actually do not contribute to the task solving. Evolving these untransited branches do not result in any reasonable improvement of g, which is defined by a name "*invalid evolution problem*" in this paper.

Beyond the invalid evolution problem, uniform genetic operators result in also the "*negative evolution*" that in a particular generation invalid evolution appears in some untransited branches to modify their alleles, while such unused nodes was phenotypically used in some previous generations. Such a situation would cause the frequent breakage of *building blocks* without notice and evaluation. This would obviously cause the potential uncertainty and low efficiency of evolution.

3.1 GNP with Simplified Genetic operators (`GNP_simplified`)

To address the above problems, this paper proposes novel genetic operators for GNP, named simplified genetic operators. The procedure of `GNP_simplified` is:

1. Initially, define the set of transited branches $b^*[g] \leftarrow \emptyset$ for each individual g;
2. FOR each individual g
   (a) add the transited branches TB_g into $b^*[g]$;
   /* **simplified crossover**: step 3 to 5 */
3. Select two parent individuals P1 and P2;
4. FOR each branch $b \in b^*[\mathrm{P1}] \bigcup b^*[\mathrm{P2}]$
   (a) IF (random_seed$< p_c$)
      i. exchange the corresponding C of b between P1 and P2;
      ii. remove b from $b^*[\mathrm{P1}]$ and $b^*[\mathrm{P2}]$;
5. go back to step 3 until crossover ends;
   /* **simplified mutation**: step 6 to 8 */
6. Select parent individual P;
7. FOR each branch $b \in b^*[\mathrm{P}]$
   (a) IF (random_seed$< p_m$)
      i. randomly set C of b at $[1, |N_{\mathrm{node}}|]$ except its corresponding node;
      ii. remove b from $b^*[\mathrm{P}]$;
8. go back to step 6 until mutation ends;
9. go back to step 2 until terminal conditions.

Intuitively, the proposed simplified genetic operators restrict the evolution only to the activated region of each individual g, i.e., $b^*[g]$, while the unused sub-graphs remain unchanged. On the other hand, once new genes are produced from the activated branches, these new branches are reset back to be untransited (step 4.(a).ii and step 7.(a).ii), because that they are new information which GNP has not confirmed its utility while evolving them will result in invalid evolution.

3.2 Discussion with Uniform Genetic Operators

Obviously, the proposed genetic operators would contribute to save time for GNP evolution, which is the reason why we call it *simplified*:

Theorem 1. *Given an individual g, the search dimension of GNP is reduced from $\sum_{i \in N_{\mathrm{node}}} |B(i)|$ to $|b^*[g]|$ via the simplified genetic operators.*

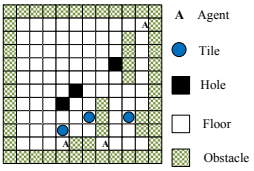


Fig. 3. Tileworld

Table 1. *LIBRARY* of Tileworld

Function	Description	#Branches
$J_1 \sim J_4$	`Judge Forward, Backward, Left, Right`	5
J_5	`Direction` of the nearest `Tile` from the agent	5
J_6	`Direction` of the nearest `Hole` from the agent	
J_7	`Direction` of the nearest `Hole` from the nearest `Tile`	
J_8	`Direction` of the `Second` nearest `Tile` from the agent	
$P_1 \sim P_4$	`Move Forward, Turn Left, Turn Right, Stay`	1

Deeper foundations motivating the proposed method arise from the nature of brain science and evolution. First, we only evolve the activated branches which GNP has confirmed their effectiveness, while the untransited region indicates some kinds of unknown skills of GNP, which could be called *intron* in Biology. Evolving these untransited parts is meaningless (invalid evolution) in nature sense, because we don't even know the quality of them, and perhaps evolving them might cause too much randomness (negative evolution). It is explicit that the simplified genetic operators can avoid the invalid and negative evolution. On the other hand, the proposed method inspires another feature: *atavism*. Evolutionarily, traits that have disappeared phenotypically actually do not disappear from the DNA, where the gene sequence often remains even it is inactive [13]. In the proposed method, the branches which have been transited in the previous generations are remained in b^* if they are not evolved in the current generation, representing the dormant genes. This keeps the experienced branches with the possibility to evolve in the future generations, resulting in a stronger guidance of evolution without frequent disruption of BBs comparing with uniform method.

4 Experimental Study

As suggested by No-Free-Lunch Theorem that there is no optimization technique to perform best results in all problems, each EA would have its own suitable problem domains. In this paper, we evaluate the performance of the simplified genetic operators in a widely-studied benchmark testbed – Tileworld [10].

Tileworld consists of a grid of cells on which agents, floors, obstacles, tiles and holes exist (Fig. 3). The agent has sensory ability to judge the environment and take the appropriate actions to move to its neighboring cells (Table 1). The primary objective of Tileworld is to find the optimal strategy that could control the agents to push tiles into holes as many and fast as possible by given limited steps. Accordingly, the evaluation of a strategy can be calculated by: $f = 100DT + 3(ST - S_{\text{used}}) + 20\sum_{t \in Tile}(D(t) - d(t))$. Here, DT is the number of tiles that have been pushed into the holes. ST is user-defined limited steps. S_{used} denotes the number of steps that have been used. $Tile$ represents the set of tiles. $D(t)$ is the original distance from tile t to its nearest hole, and $d(t)$ is the distance from t to its nearest hole after ST steps.

In this paper, ten Tileworlds consisting of 3 agents, tiles and holes randomly positioned in Fig. 3 are used. In every step, 3 agents are controlled for movement

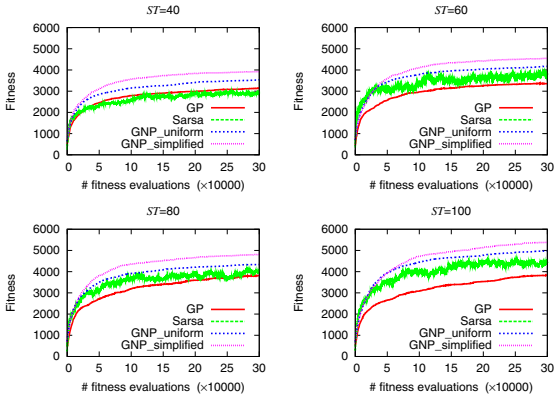


Fig. 4. Fitness curves of Tileworld under different ST

Table 2. Detailed results

	Fitness	t-test
$ST = 40$		
GP	3139.9 ± 482.3	**6.54e-6**
Sarsa	2971.7 ± 492.5	**9.73e-7**
GNP_uniform	3532.1 ± 540.4	**1.18e-2**
GNP_simplified	**3920.1 ± 548.3**	—
$ST = 60$		
GP	3356.3 ± 679.2	**8.08e-8**
Sarsa	3719.2 ± 1106.2	**1.34e-3**
GNP_uniform	4171.4 ± 692.0	**4.23e-2**
GNP_simplified	**4547.3 ± 596.8**	—
$ST = 80$		
GP	3802.5 ± 789.8	**8.06e-5**
Sarsa	4031.1 ± 802.2	**1.21e-3**
GNP_uniform	4337.0 ± 773.4	**2.79e-2**
GNP_simplified	**4816.9 ± 912.4**	—
$ST = 100$		
GP	3830.0 ± 1080.9	**1.93e-7**
Sarsa	4463.1 ± 809.3	**5.56e-4**
GNP_uniform	4974.3 ± 556.1	**4.37e-2**
GNP_simplified	**5381.9 ± 878.3**	—

simultaneously. The objective of evolution is to find the optimal solution with highest f in ten worlds, hence the fitness calculation $Fitness = \sum_{w=1}^{10} f(w)$.

4.1 Parameter Configuration

The overall parameters of GNP include: population size is 300 (1 for elite preservation, 120 for crossover and 179 for mutation); the maximal number of fitness evaluations is 300,000.

The directed graph G is defined according to [5], where $N_{\text{node}} = 60$. Different ST is set to testify the scalability with different problem difficulty. With the decrease of ST, the problem difficulty tends to increase. In this paper, $ST = \{40, 60, 80, 100\}$. The best $p_c = 0.1$ and $p_m = 0.02$ are performed by hand-tuning. The other compared state-of-the-art algorithms include original `GNP_uniform`, GP [14] and Sarsa Learning [15], whose parameters are set based on [5]. All the presented experimental results are the average of 30 independent trials.

4.2 Experimental Study

Fitness Results and Scalability: The fitness curves are plotted in Fig. 4. It is found that the proposed `GNP_simplified` outperforms GP, Sarsa and the original `GNP_uniform` among all tested 4 problems. The detailed results and their statistical t-tests are given in Table 2, showing the superiority of `GNP_simplified` with statistical meaning.

Evolution Behavior: To understand the advantages of the simplified genetic operators, an intuitive analysis of its evolution behavior is carried out. As Fig. 2 illustrated, each GNP individual only transits a part of its directed graph to adapt to the detected environment. Accordingly, when some unused branches

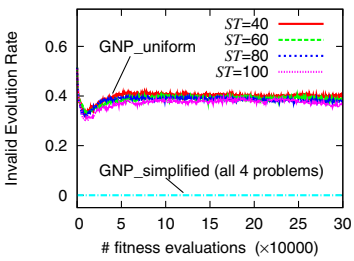


Fig. 5. Invalid evolution rates

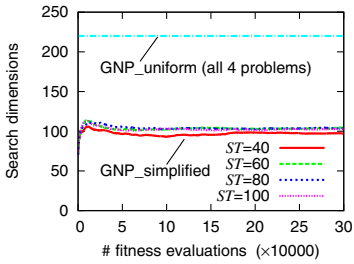


Fig. 6. Change of search dimensions

are taken into account for evolution, invalid evolution will happen. Fig. 5 plots the invalid evolution rates (defined by the fraction that evolution appears in the branches $\notin b^*$) of `GNP_uniform` changed over generations in all 4 problems, while `GNP_simplified` never causes the invalid evolution. Fig. 6 shows the changes of search dimensions during evolution. In each generation, `GNP_simplified` only evolve the activated regions of G, which is adaptively determined by the interactions between the detected environments and individual executions. On the other hand, original `GNP_uniform` considers all branches as the search dimensions, ignoring the fact of invalid evolution. Consequently, `GNP_simplified` can reduce the search dimensions, which biases the evolution power towards recombining the experienced sub-structures, allowing higher evolution efficiency.

4.3 Discussions

Overall, the proposed `GNP_simplified` is capable of obtaining higher fitness values comapring with the original `GNP_uniform` and some state-of-the-art algoirhtms in the benchmark of Tileworld with different problem complexities. In the intuitive analysis of evolution behavior, it is found that even with different problem complexities, original `GNP_uniform` tends to have similar level of invalid evolution rates, proving that the invalid evolution problem is widespread in GNP. On the other hand, with the proposal of `GNP_simplified`, the search dimensions of GNP can be stably reduced, i.e., almost all reduced to a half-level in 4 test problems. We present the explanation of the superiority of the simplified genetic operators in mainly two reasons: First, `GNP_simplified` can avoid the invalid evolution problem; Second, `GNP_simplified` reduces the search dimensions of the directed graph, which contributes to simplify the search space of evolution.

5 Conclusions and Future Work

In this paper, simplified genetic operators for GNP are proposed. None of traditional genetic operators considers the unique features of GNP, which tends to cause unnecessary invalid evolution problem. The proposed simplfied genetic operators explicitly consider the “transition by necessity” feature of GNP to only

evolve the activated regions of GNP. Our study demonstrates the superiority of the proposed method in terms of avoding the invalid evolution and reducing the search dimensions. Since the aim of this study is to propose general-suitable genetic operators for GNP, in the future we will try to extend this work to address the real-world applications.

References

1. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. 1(1), 67–82 (1997)
2. Teller, A., Veloso, M.: PADO: Learning tree structured algorithms for orchestration into an object recognition system. Tech. Report CMU-CS-95-101, Carnegie Mellon University (1995)
3. Hirasawa, K., Okubo, M., Katagiri, H., Hu, J., Murata, J.: Comparison between genetic network programming (GNP) and genetic programming (GP). In: Proc. of the IEEE Congress on Evol. Comput., pp. 1276–1282 (2001)
4. Mabu, S., Hirasawa, K., Hu, J.: A graph-based evolutionary algorithm: Genetic network programming (GNP) and its extension using reinforcement learning. Evol. Comput. 15, 369–398 (2007)
5. Li, X., Mabu, S., Hirasawa, K.: A novel graph-based estimation of distribution algorithm and its extension using reinforcement learning. IEEE Trans. Evol. Comput. (early access, 2013)
6. Hirasawa, K., Eguchi, T., Zhou, J., Yu, L., Markon, S.: A double-deck elevator group supervisory control system using genetic network programming. IEEE Trans. Syst., Man, Cybern. C 38, 535–550 (2008)
7. Li, X., Mabu, S., Zhou, H., Shimada, K., Hirasawa, K.: Genetic network programming with estimation of distribution algorithms for class association rule mining in traffic prediction. In: Proc. of the IEEE Congress on Evol. Comput., pp. 2673–2680 (2010)
8. Li, X., Mabu, S., Hirasawa, K.: An extended probabilistic model building genetic network programming using both of good and bad individuals. IEEJ Trans. on Electrical and Electronic Engineering 8, 339–347 (2013)
9. Li, X., Hirasawa, K.: Extended rule-based genetic network programming. In: Proc. of the Genetic and Evol. Comput. Conf. Companion, pp. 155–156 (2013)
10. Pollack, M.E., Ringuette, M.: Introducing the tile-world: Experimentally evaluating agent architectures. In: Proc. of the AAAI, pp. 183–189 (1990)
11. Hirasawa, K., Ohbayashi, M., Sakai, S., Hu, J.: Learning Petri network and its application to nonlinear system control. IEEE Trans. Syst., Man, Cybern. B 28(6), 781–789 (1998)
12. Katagiri, H., Hirasawa, K., Hu, J., Murata, J.: Comparing some graph crossover in genetic network programming. In: Proc. of the SICE Conf., pp. 1263–1268 (2002)
13. Collin, R., Cipriani, R.: Dollo's law and the re-evolution of shell coiling. Royal Society B 270(1533), 2551–2555 (2003)
14. Koza, J.R.: Genetic Programming, on the Programming of Computers by Means of Natural Selection. MIT Press (1992)
15. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (1998)

Hierarchical Clustering for Local Time Series Forecasting

Aymen Cherif[1], Hubert Cardot[1], and Romuald Boné[2,1]

[1] Université François Rabelais Tours
Laboratoire d'informatique
64 avenue Jean Portalis, 37200 Tours, France
[2] Ecole Nationale d'Ingénieurs du Val de Loire
3, Rue de la Chocolaterie, BP 3410, 41034 Blois cedex, France
{aymen.cherif,hubert.cardot,romuald.bone}@univ-tours.fr

Abstract. We are interested in local models for time series forecasting and we propose a new approach based on hierarchical clustering. This new approach uses a binary tree, k-means clustering and pruning strategies to find out the adequate clustering with MLP and SVM as predictors. The experimentations give very good results for this approach. Three strategies were tested and a comparative study with other methods show that the hierarchical predictor model outperformed the existing models on the three used datasets.

Keywords: Multilayer perceptron, SVM (Support Vector Machines), time series forecasting, local model, hierarchical clustering, binary trees, k-means.

1 Introduction

Time series forecasting is a widely discussed issue which is found in many domains. Our focus in this paper will be on local models for time series forecasting. The objective of these models can be described in three steps: during the first step, the time series embedded into M-dimensional space vectors, then they are clustered into sub learning sets and finally local predictions are executed on each subset.

In section 2, a brief presentation of related works in local approaches are presented. In section 3, we describe our hierarchical model, followed by the experimentations on section 4 and finally we conclude with a discussion of this methods and possible perspectives in section 5.

2 Related Works

In the literature, many models and algorithms have been proposed, among them are, Multi-Layer Perceptron (MLP) and Support Vector Machines (SVM). In addition, methods and techniques to push the limits of prediction error have

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 59–66, 2013.

been proposed. The local model [1,2] techniques are one of them. It consists in dividing the data into homogeneous clusters using Vector Quantization (VQ [3]).

Theoretically, dividing the data in distinct and homogeneous clusters should allow the predictor to estimate an easier function in each cluster instead of a global and probably more complex one. For example in [1], this technique was used to predict non-linear time series by linear predictors. Another application was to predict seasonal time series in [4].

After embedding the time series values into a multi-dimensional vector space which is done by sliding a temporal window on the data sets, comes the step of dividing the data into subspaces or clusters. According to the literature several methods were used for this task. In our knowledge, all of them come from the category of partitional clustering. The most used method is the Self-Organizing Maps (SOM [5]) and the Neural Gas (NG [6]). Many adaptations for temporal sequences were presented in [7].

The third and final step is the prediction where for each created cluster a predictor is associated during the learning process. Many types of predictors were used such as MLP [2] and SVM [8]. All these methods share the point that the prediction and clustering steps are not linked. Our proposition tries to emphasize this particular point in order to improve the final results.

3 Proposed Method

We propose a new approach for time series forecasting which is based on the same principle of local models seen in the previous section. According to the literature most of the local approaches are based on SOM clustering algorithms and more generally on partitional clustering algorithms. Both experiments and rational deduction show that the number of clusters and their composition are important factors for forecasting performances. Since existing methods do not propose any link between the clustering step and the prediction step, we propose a new hierarchical local model that is based on binary tree for clustering data. The idea is to drive the tree development by the local predictor performances.

Our method tends to provide a link between the prediction step and the clustering step which is missing in previous works. Further more, the introduction of hierarchical architecture allows us to imagine several parameters in order to control the height of the tree. For example, the cluster size is useful to avoid small clusters and limits the number of nodes.

The tree development is performed through k-means algorithm. Basically, for each node, one 2-means with two centroids (k=2) is applied on vectors linked to this node. The k-means choice is justified by fastness and simplicity. We have chosen a binary tree modelling (k=2) to start with the simplest way.

In this model the predictors are used as black boxes. They take as input a temporal window which represents the past values and they give as output the corresponding future value. Typically, each node has his own predictor in which the evaluation of efficiency of the clustering is done by the normalized mean square error (NMSE) over a validation set.

3.1 Tree Description

We describe a binary tree as a graph defined by :

$$T = (V, E, \gamma) \tag{1}$$

where V denotes the set of nodes, E the set of oriented edges linking two nodes and γ is the function that links two nodes v_1 and v_2 by the edge a defined by

$$\begin{aligned} \gamma : E &\to V \times V \\ a &\mapsto (v_1, v_2) \end{aligned} \tag{2}$$

Three node types can be defined : root node, leaf nodes and inner nodes. The root node is unique in a tree and have only two edges toward only child nodes. The leaves are the tree bottom nodes with no child. We refer by inner nodes the ones in the middle (they have child nodes and a parent node).

For each node v_i are associated a set of vectors from the learning set A denoted by $A(v_i)$ which can be seen as the obtained clusters at the node v_i. We define an horizontal cut H for a tree T as the clustering of the set A verifying the two conditions

$$A = \bigcup_{i=1}^{k} A(v_i) \; and \; \bigcap_{i=1}^{k} A(v_i) = \emptyset \tag{3}$$

where k is the number of nodes in H. For example, the root node r and the leaf set can be seen as two horizontal cuts for T.

This approach opens the possibility for different strategies to find an adequate horizontal cut. For instance, when the time processing is important, we can use a strategy that stops early the tree decomposition. From an opposite point of view we can think about a strategy that construct the complete tree and then, with a pruning technique, try to find out the optimal clustering.

We propose in this paper three different strategies.

3.2 Strategy 1 : Pre-pruning

In this strategy, tree development is done alternatively with the predictors evaluation. In this way the pruning is applied during the tree construction, for this reason we call this strategy pre-pruning. Our proposition is to develop each node at one step for two successive levels. This leads to one complete tree of two levels ($h = 2$) with one root node, two regular nodes and four leaves, respectively r, n_1, n_2, l_i $(1 \leq i \leq 4)$, as shown in the figure 1. Then the predictors are applied on these 7 nodes. In this configuration 5 horizontal cuts are possible represented in the figure 1.

The proposed algorithm uses one list L_c that contains the leaf nodes to be developed and stops when L_c is empty. The algorithm 1 describes the principles of the operation. $\mathcal{F}$ denotes the leaf sets and $find\_cut\_min(n)$ is a function that return the optimal cut for the sub tree exposed in figure 1 by the five different colors.

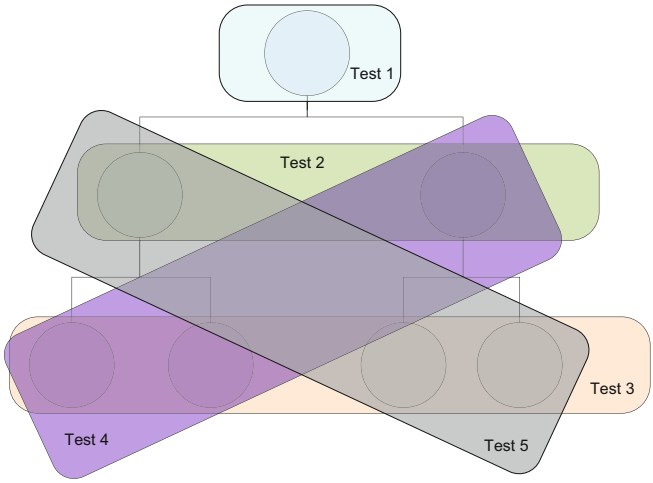


Fig. 1. Obtained tree by pre-pruning strategy, each color represents a different horizontal cut

Algorithm 1. Pre-pruning algorithm

```
L_c ← r
while ¬empty(L_c) do
  n ← unstack(L_c)
  Develop(n)
  Predict(n)
  H ← find_cut_min(n)
  for all m ∈ H do
    if m ∈ F then
      L_c ← m ∈ H
    else
      L_f ← m ∈ H
    end if
  end for
end while
return L_f
```

3.3 Strategy 2 : Post-pruning

In this strategy, the idea is to develop the tree at its maximum ability. It operates in three steps: the development step which is the tree construction, the learning step where all predictors perform the learning and finally selecting clustering by searching the best horizontal cut over a validation set.

The pruning in this strategy is performed by an $O(n)$ algorithm (where n is the number of nodes). The algorithm uses two lists : L_f for the final horizontal cut and a temporary list L_t.

In the beginning, the final list can be initialized by any horizontal cut, we choose the set of leaves. In each iteration one node is selected and replaces all its progeny in the temporary list. By repeating this, we save the horizontal cut that gives the best performance in the validation set. Detailed steps can be found in the algorithm 2.

Algorithm 2. Post-pruning algorithm

$L_f \leftarrow \mathcal{F}$
$MSE \leftarrow calculateGlobalError(L_f)$
for all $v_i \in \mathcal{V} - \mathcal{F}$ **do**
  $L_t \leftarrow L_f - progeny(v_i) + \{v_i\}$
  $MSE_t \leftarrow calculateGlobalError(L_t)$
  if $MSE_t < MSE$ **then**
    $MSE \leftarrow MSE_t$
    $L_f \leftarrow L_t$
  end if
end for
return L_f

3.4 Strategy 3 : Leaf Mapping

In this strategy, we do not search for the optimal clustering from the tree. The idea is to take benefits from all trained predictors in the tree. In order to do that we define for each node n_i its local prediction error E_i and the cluster size denoted τ_i, defined by :

$$\tau_i = \frac{Card(n_i)}{Card(r)} \tag{4}$$

where $Card(n)$ is the number of vectors affected to the node n.

This allow us to establish the error contribution for each node by the expression :

$$c_i = \tau_i \times E_i \tag{5}$$

It is easy to establish the relation between the global prediction error and the contributions :

$$NMSE = \sum_{n_i \in H} c_i \tag{6}$$

from the observation that the c_i are positif elements, we may think that minimizing the global error could be assimilated to finding the nodes that give the less contributions. In the proposed algorithm, we associate to each leaf the node that has the smallest error contribution from the parents (including the node itself).

4 Experimentation

In this section we test the proposed methods on well-known time series (Sunspots, Mackey-Glass and FIR-Laser). For comparison purposes, we studied earlier works like [9,2,1,10] which have used the same data under similar conditions. In order to see clearly the contribution of the hierarchical models on the results, both MLP and SVM were used as predictors and we used the Normalized Mean Square Error (NMSE) criteria.

4.1 Tests on Sunspots

State of the art results for Sunspot time series are presented in the table 1. The results are compared with a recurrent neural networks (RNN) in [11], an MLP with feature selection in [12], a boosted RNN in [13] and SVM predictor with composite kernel [9]. The best results are obtained by the mapping strategy + MLP predictor.

Table 1. Sunspot compared results

Model	NMSE Test
RNN [11]	0.084
MLP [12]	0.078
Boosting [13]	0.078
SVM [9]	0.039
Mapping + MLP	0.0034

4.2 Tests on Laser

The table 2 summaries state of the art results for Laser time series. in this table we can find implementation of RSOM local model in [10] using MLP predictors. In [14] a MLP using FIR connections is used. In [15] another local model is presented based on SOM and SVM. A RNN with an algorithm of adding connections is proposed in [11]. And finaly, in [13] a boosted RNN is used. We can see that our proposed method, based on Pre-Pruning + MLP, gives the best results over all.

4.3 Tests on MG-17

Table 3 presents some results on MG-17, in wich we found a local model of [2] with SOM and MLP and [15] with SOM and functional networks and the boosted RNN [13]. It's shown that our model outperform other methods for MG-17 too, the best performances are obtained by pre-pruning strategy + SVM predictor.

The results exposed in the table 4 show best performances obtained by the three strategies with the three time series. Both, MLP and SVM as predictors are presented. For Sunspot, the mapping strategy gives the best results by using

Table 2. Laser compared results

Model	NMSE Test
RSOM [10]	0.084
FIR MLP [14]	0.023
[15]	0.011
RNN [11]	0.008
Boosting [13]	0.005
Pre-Pruning + MLP	0.0039

Table 3. MG-17 compared results

Model	NMSE Test (10^{-3})
Barreto [16]	0.75
Vesanto [2]	0.17
Boosting [13]	0.11
NoeliaSanchez [15]	5.75×10^{-4}
Pre-Pruning + SVM	1.37×10^{-5}

MLP predictor (0.0034) and SVM with (0.0045). However, the three stategies are close. For Laser, the best perfomance is obtained by pre-pruning with both MLP and SVM predictors (respectively 0.0039 and 0.0043). For MG-17 time series, the superiority of the SVM is clear and confirmed by post-pruning and pre-pruning strategies with equal NMSE (1.37×10^{-8} for SVM and 9.82×10^{-6} for MLP).

Table 4. Best performances obtained by hierarchical models

	SunSpots		Laser		MG-17	
	MLP	SVM	MLP	SVM	MLP	SVM
Pre Prunning	$0,0040$	$0,0049$	**0,0039**	$0,0043$	$9,82E-06$	**1,37E-08**
Post Pruning	$0,0051$	$0,0046$	**0,0039**	$0,0050$	$9,82E-06$	**1,37E-08**
Mapping	**0,0034**	$0,0045$	$0,0040$	$0,0043$	$1,50E-05$	$1,94E-08$

5 Discussion

According to the experimentations, it appears that our proposition outperforms the existing methods but also put classical predictors such as MLP or SVM to a better position than more complicated ones (for example RNN). This improvement can be linked to one important fact : clustering step and prediction step are no longer separated. Not only the number of clusters can be "self-controlled" but also the tree composition. As future work, it can be suggested to find a relation between the hierarchical clustering and the best pruning strategy to use.

References

1. Walter, J., Riter, H., Schulten, K.: Nonlinear prediction with self-organizing maps. In: 1990 International Joint Conference on, pp. 589–594 (1990)
2. Vesanto, J.: Using the SOM and local models in time-series prediction. In: Proc. Workshop on Self-Organizing Maps, pp. 209–214 (1997)
3. Gersho, A., Gray, R.M.: Vector Quantization and Signal Compression. The Kluwer international series in engineering and computer science: Communications and information theory: Kluwer international series in engineering and computer science: Kluwer international series in engineering and computer science. Kluwer (1992)

4. Martín-Merino, M., Román, J.: A New SOM Algorithm for Electricity Load Forecasting. In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4232, pp. 995–1003. Springer, Heidelberg (2006)
5. Kohonen, T., Oja, E.: Visual feature analysis by the self-organising maps. Neural Computing & Applications 7(3), 273–286 (1998)
6. Martinetz, T.M., 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(4), 558–569 (1993)
7. Cherif, A., Cardot, H., Boné, R.: SOM time series clustering and prediction with recurrent neural networks. Neurocomputing 74(11), 1936–1944 (2011)
8. Cao, L.: Support vector machines experts for time series forecasting. Neurocomputing 51, 321–339 (2003)
9. Jiang, T., Wang, S., Wei, R.: Support vector machine with composite kernels for time series prediction. In: Liu, D., Fei, S., Hou, Z., Zhang, H., Sun, C. (eds.) ISNN 2007, Part III. LNCS, vol. 4493, pp. 350–356. Springer, Heidelberg (2007)
10. Koskela, T., Varsta, M., Heikkonen, J., Kaski, K.: Time Series Prediction Using Recurrent SOM with Local Linear Models. Int. J. of Knowledge-Based Intelligent Engineering Systems 2, 60–68 (1997)
11. Bone, R., Crucianu, M., Asselin de Beauville, J.-P.: Two constructive algorithms for improved time series processing with recurrent neural networks. In: Neural Networks for Signal Processing X. Proceedings of the 2000 IEEE Signal Processing Society Workshop (Cat. No.00TH8501), pp. 55–64 (2000)
12. Czernichow, T.: Apport des réseaux récurrents à la prévision de séries temporelles, applications à la prévision de consommation d'électricité. PhD thesis, Université Paris 6, Paris (1993)
13. Assaad, M., Bone, R., Cardot, H.: A new boosting algorithm for improved time-series forecasting with recurrent neural networks. Information Fusion 9(1), 41–55 (2008)
14. Wan, E.A.: Finite impulse response neural networks for autoregressive time series prediction. Time Series Prediction: Forecasting the Future and Understanding the Past (2), 195–218 (1993)
15. Sánchez-Maroño, N., Fontenla-Romero, O., Alonso-Betanzos, A., Guijarro-Berdiñas, B.: Self-organizing maps and functional networks for local dynamic modeling. In: Proceedings of the European Symposium on Artificial Neural Networks (ESANN 1995), pp. 39–44 (2003)
16. Barreto, G.A., Araujo, A.F.R.: Identification and control of dynamical systems using the self-organizing map. IEEE Transactions on Neural Networks 15(5), 1244–1259 (2004)

Multi-regularization for Fuzzy Co-clustering

Vikas K. Garg[1], Sneha Chaudhari[2], and Ankur Narang[2]

[1] Toyota Technological Institute, Chicago
vkg@ttic.edu
[2] IBM Research Lab, India
{snechaud,annarang}@in.ibm.com

Abstract. Co-clustering is a powerful technique with varied applications in text clustering and recommender systems. For large scale high dimensional and sparse real world data, there is a strong need to provide an overlapped co-clustering algorithm that mitigates the effect of noise and non-discriminative information, generalizes well to the unseen data, and performs well with respect to several quality measures. In this paper, we introduce a novel fuzzy co-clustering algorithm that incorporates multiple regularizers to address these important issues. Specifically, we propose *MRegFC* that considers terms corresponding to Entropy, Gini Index, and Joint Entropy simultaneously. We demonstrate that MRegFC generates significantly higher quality results compared to many existing approaches on several real world benchmark datasets.

1 Introduction

Co-clustering or bi-clustering is a powerful tool that alleviates notable limitations of clustering techniques such as poor scalability, lack of cluster intrepretability and sensitivity to noise [1]. Co-clustering allows simultaneous clustering of the rows and columns of a matrix, and has been used successfully in text mining [2], [3] and collaborative filtering [4]. In collaborative filtering, for example, co-clustering can be used for identifying groups of customers with similar interests or preferences toward a set of products. The co-clusters thus obtained can be leveraged for target marketing in recommender systems.

Many co-clustering methods partition the data into non-overlapping regions where each point belongs to only one cluster such as ITCC [3], Bregman co-clustering [5]. However, in real world applications, *fuzzy* co-clustering, that allows the data points to be members of two or more clusters, is more suitable. For example, when clustering documents into topics, documents may contain multiple relevant topics and hence an overlapped co-clustering is more appropriate [6]. Overlapped co-clustering algorithms also capture the vague boundaries between clusters and improve the representation and interpretability of the clusters.

Further, certain issues need to be addressed for obtaining superior performance using fuzzy co-clustering. The points or features occurring across a large number of clusters should not be allowed to dominate since they contain very little discriminative information. Also, noise in the underlying data needs to be

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 67–75, 2013.

effectively handled. One common way to deal with such issues is to devise fuzzy techniques that focus on optimizing an objective based on some regularizer as shown in FCR [7] and FCM with Maximum Entropy regularization [8]. A major limitation of these techniques lies in the insufficiency of a single regularizer to perform well with respect to several quality measures. For example, FCR uses entropy in the objective function which helps to obtain better degree of aggregation on real datasets, but shows lower accuracy.

FCCM [9] is a fuzzy clustering algorithm that maximizes the co-occurrence of categorical attributes (keywords) and the individual patterns (documents) in clusters. However, this algorithm poses difficulties while handling large data sets and also works for only categorical data. *Fuzzy-CoDoK* [10], a scalable modification of FCCM, involves heavy parameter tuning that makes the approach data-dependent, is susceptible to variations in data and may often fail to converge. Technique such as *SCAD* [11] only works with data lying in some Euclidean space. *SKWIC* [12] overcomes this limitation but lacks in parameter tuning and scalability. Similarly, technique such as *MOCC* [13] performs poorly with respect to degree of aggregation.

In this paper, we formulate a framework, Multi-Regularization for Fuzzy Co-clustering (MRegFC), based on maximizing an objective function that incorporates penalty terms based on the Entropy, the Gini Index, and the Joint Entropy simultaneously under certain constraints. Each one of the regularizers used in MRegFC contribute to address the issues related to co-clustering, as explained later in Section 2. MRegFC can also handle high dimensional and sparse data without over-fitting. However, incorporating multiple regularizers becomes challenging as different regularizers might have contrasting behaviors and learning a good set of weights for several regularizers simultaneously is important. Our technique MRegFC alleviates both these issues. Further, MRegFC provides valid range of values for different parameters used, to obtain high quality results. In experimental evaluation, we demonstrate superior performance in terms of precision, recall, and F-measure as compared to prior approaches: MOCC [13], ITCC [3], FCR [7] and algorithms employing only one of these regularizers. Our algorithm also demonstrates better RMSE compared to FCR [7] and individual regularizers on all the datasets in consideration. To the best of our knowledge, MRegFC is the first multiple regularizer based approach for fuzzy co-clustering.

2 The Proposed Approach

In this work we propose an approach called MRegFC for fuzzy co-clustering which formulates the objective function employing the Entropy, the Gini Index, and the Joint Entropy regularizers simultaneously. A regularization term is added to the objective function in order to prevent it from being an ill-posed problem and to avoid overfitting. The regularization term based on Entropy [8] elegantly captures the notion of *purity* of a co-cluster while emphasizing the marginal coherence along the rows (points) and the columns (features). Hence, the homogeneity along the points and the features are appropriately taken into

Table 1. Notation

Symbol	Definition
C	Number of co-clusters
N	Number of data points (rows)
K	Number of features (columns)
u_{ci}	Membership of row i in co-cluster c
v_{cj}	Membership of column j in co-cluster c
d_{ij}	Measure of extent of correlation between row i and column j

account using the Entropy regularizer. Gini Index, despite being similar to Entropy, ensures that the points and features that occur across a large number of clusters are not provided with any unfair advantage, besides imparting numerical stability to the algorithm [10]. Joint Entropy [14] characterizes, in a natural way, the statistical dependence of the points (rows) and the features (columns) on each other. Moreover, the Joint Entropy term, in conjunction with Entropy, creates a Mutual Information term thereby lending a better generalization ability to MRegFC by making it robust against noise. It is easy to see that the Joint Entropy term is maximized when the product $u_{ci}.v_{cj}$ is evenly distributed across the different co-clusters. Thus incorporating a joint entropy fuzzifier also reduces the susceptibility of the algorithm to overfitting[1].

A typical fuzzy co-clustering algorithm strives to maximize an objective function, generally the degree of aggregation. Using the notations given in Table 1, the degree of aggregation for cluster c can be quantified as

$$\sum_{i=1}^{N}\sum_{j=1}^{K} u_{ci}v_{cj}d_{ij}, \quad \text{for } c \in \{1, 2, \ldots, C\} \tag{1}$$

The intuition is that we want to bring together rows and columns with high d_{ij} values in the same co-cluster. To maximize the value of the objective function, for such i and j, we need to set high values for both u_{ci} and v_{cj} for the same cluster c. Additionally, we impose the following constraints:

$$\sum_{c=1}^{C} u_{ci} = 1, u_{ci} \in [0, 1], i \in \{1, 2, \ldots, N\} \tag{2}$$

$$\sum_{j=1}^{K} v_{cj} = 1, v_{cj} \in [0, 1], c \in \{1, 2, \ldots, C\} \tag{3}$$

The first constraint requires that the addition of membership values of each row across all the co-clusters is equal to 1. Such a constraint is said to satisfy the *Ruspini's condition* [15]. The second constraint, on the other hand, requires that the summation of all column memberships must be one for each co-cluster

[1] The under-fitting issues are implicitly taken care of by the term corresponding to the degree of aggregation.

thereby implying a weighting scheme for the columns, instead of the partitions[2]. We now add regularization terms corresponding to Entropy, Gini Index and Joint Entropy in the objective function. Consequently, using the weight parameters T_{u_1} and T_{u_2} for Entropy, T_{v_1} and T_{v_2} for Gini Index, T_{uv} for Joint Entropy to specify the extent of fuzziness, we strive to maximize our regularized objective function, OBJ

$$
\begin{aligned}
&= \sum_{c=1}^{C}\sum_{i=1}^{N}\sum_{j=1}^{K} u_{ci}v_{cj}d_{ij} - T_{uv}\sum_{c=1}^{C}\sum_{i=1}^{N}\sum_{j=1}^{K} u_{ci}v_{cj}\log(u_{ci}v_{cj}) - T_{u_1}\sum_{c=1}^{C}\sum_{i=1}^{N} u_{ci}\log(u_{ci}) \\
&- T_{u_2}\sum_{c=1}^{C}\sum_{i=1}^{N} u_{ci}^2 - T_{v_1}\sum_{c=1}^{C}\sum_{j=1}^{K} v_{cj}\log(v_{cj}) - T_{v_2}\sum_{c=1}^{C}\sum_{j=1}^{K} v_{cj}^2 \\
&+ \sum_{i=1}^{N}\lambda_i\left(\sum_{c=1}^{C} u_{ci} - 1\right) + \sum_{c=1}^{C}\gamma_c\left(\sum_{j=1}^{K} v_{cj} - 1\right) \qquad (4)
\end{aligned}
$$

Differentiating with respect to u_{ci} we get

$$
\Rightarrow \frac{\partial OBJ}{\partial u_{ci}} = \sum_{j=1}^{K} v_{cj}d_{ij} - T_{u_1}\left(1+\log(u_{ci})\right) - 2T_{u_2}u_{ci} - T_{uv}\sum_{j=1}^{K} v_{cj}\left(1+\log(u_{ci}v_{cj})\right) + \lambda_i \qquad (5)
$$

For optimality of OBJ, we must have $\dfrac{\partial OBJ}{\partial u_{ci}} = 0$. Further, since $u_{ci}, v_{cj} \in [0,1]$, approximating $(1+\log u_{ci})$ and $(1+\log u_{ci}v_{cj})$ by u_{ci} and v_{cj} respectively, we have

$$
\Rightarrow u_{ci} = \frac{\lambda_i + \sum_{j=1}^{K} v_{cj}d_{ij}}{T_{u_1} + 2T_{u_2} + T_{uv}\sum_{j=1}^{K} v_{cj}^2} \qquad (6)
$$

Now using $\sum_{c=1}^{C} u_{ci} = 1$, and simplifying, we obtain

$$
u_{ci} = \frac{1}{C} + \frac{1}{T_{u_1} + 2T_{u_2} + T_{uv}\sum_{j=1}^{K} v_{cj}^2} * \left(\sum_{j=1}^{K} v_{cj}d_{ij} - \frac{1}{C}\sum_{t=1}^{C}\sum_{j=1}^{K} v_{tj}d_{ij}\right) \qquad (7)
$$

Following a similar procedure of obtaining u_{ci}, we can compute

$$
v_{cj} = \frac{1}{K} + \frac{1}{T_{v_1} + 2T_{v_2} + T_{uv}\sum_{i=1}^{N} u_{ci}^2} * \left(\sum_{i=1}^{N} u_{ci}d_{ij} - \frac{1}{K}\sum_{t=1}^{K}\sum_{i=1}^{N} u_{ci}d_{it}\right) \qquad (8)
$$

[2] We do not impose Ruspini's condition on the columns since then a single co-cluster containing all the rows and columns would be formed.

A good selection of the parameters T_{u_1}, T_{u_2}, T_{v_1}, and T_{v_2} can be mathematically derived in a straightforward way (omitted due to space constraints):

$$0 < T_{u_1} < \frac{\sum_{t=1}^{C}\sum_{j=1}^{K} v_{tj}P_j}{N} - T_{uv}\max_c \sum_{j=1}^{K} v_{cj}^2 \tag{9}$$

$$T_{u_2} = \frac{\sum_{t=1}^{C}\sum_{j=1}^{K} v_{tj}P_j - NT_{uv}\max_c \sum_{j=1}^{K} v_{cj}^2 - NT_{u_1}}{2N} \tag{10}$$

$$0 < T_{v_1} < \frac{\sum_{j=1}^{K} P_j}{C} - T_{uv}\max_c \sum_{i=1}^{N} u_{ci}^2 \tag{11}$$

$$T_{v_2} = \frac{\sum_{j=1}^{K} P_j - CT_{v_1} - CT_{uv}\max_c \sum_{i=1}^{N} u_{ci}^2}{2C} \tag{12}$$

Please note that this is a lateral benefit of our approach since in general, tuning the input parameters appropriately is a difficult problem, and the algorithm may not perform satisfactorily in the absence of any tuning guidelines.

Algorithm 1 describes our approach for fuzzy co-clustering. The algorithm takes as input the number of co-clusters C, the row-column correlation matrix D, and a threshold ϵ to specify the stopping criterion. It can be observed that the parameters λ and γ do not play a role in the resulting algorithm and hence show no effect on the overall performance. The different row memberships are randomly initialized subject to the constraint that their summation is equal to 1. Based on selection of T_{uv}, the values of the parameters T_{v1} and T_{v2} is chosen from the respective acceptable range. The algorithm then alternately updates the row and column memberships repeatedly, until the change in all the row memberships across two successive iterations is bounded by ϵ. At termination, the algorithm outputs appropriate row and column memberships across the different co-clusters.

3 Experimental Evaluation

In this section, we present experimental evaluation on several benchmark datasets that demonstrates a superior performance of *MRegFC* over the *FCR*, *ITCC* and *MOCC* algorithms. We also demonstrate the benefits of using multiple regularizers in *MRegFC* by presenting a comprehensive evaluation against the individual regularizers.

Algorithm 1. Multi-Regularized Fuzzy Co-clustering (MRegFC)

Input : No. of co-clusters C, row-col matrix D, and threshold parameter ϵ
Output: Membership values u_{ci} and v_{cj}

Compute $P_j = \sum_{i=1}^{N} d_{ij}$. Initialize randomly memberships $u_{ci} \geq 0$, $c \in [C]$ and $i \in [N]$ such that $\sum_{c=1}^{C} u_{ci} = 1$.

Choose $T_{uv} \in \left(0, \frac{\sum_{j=1}^{K} P_j}{C \max_c \sum_{i=1}^{N} u_{ci}^2} \right)$.

Choose T_{v_1} using Eqn. (11).

Compute T_{v_2} using Eqn. (12).

Compute memberships v_{cj} using Eqn. (8).

Choose T_{u_1} using Eqn. (9).

Compute T_{u_2} using Eqn. (10).

$u_{ci}^{old} \leftarrow u_{ci}$

Update memberships u_{ci} using Eqn. (7).

if $\left(\max_c |u_{ci} - u_{ci}^{old}| > \epsilon \right)$ **then**

    Update memberships v_{cj} using Eqn. (8).

    Go to step 9

end

We conducted experimentation on the following datasets [16], [13]: (a) *Movielens* for movie recommendations, (b) *Classic3* for document collections, (c) *Jester* for joke ratings (d) *Reuters (21578)* for text categorization, and (e) *20 Newsgroups* for text classification and clustering. We used two subsets of the Movielens dataset: (*a*) (*Mv1*: 679 movies from 3 genres - Animation, Children and Comedy, and (*b*) *Mv2*: 232 movies from 3 genres - Thriller, Action and Adventure. These are similar to the ones used in [13], and therefore provide for a consistent comparison with the *MOCC* and *ITCC* algorithms. Each reported result is based on an average over 10 trials. The number of clusters chosen for experiments $E1$ and $E2$ were 8 and 16, respectively for *MRegFC*; other algorithms were represented by $(5, 5)$ and $(10, 10)$ row and column clusters. The threshold parameter ϵ was set to 0.00001. In order to compare the quality of clustering results, we use the following standard measures: RMSE, precision, recall, and F-measure [13].

3.1 Comparison with Existing Approaches

Table 2 presents the comparative results for precision, recall and f-measure on the Movielens dataset. *MRegFC* has a high average precision value of around 0.73, and consistently outperforms the other algorithms. *MRegFC* achieves a high recall value of about 0.67 on an average. Further, it can be seen that *MRegFC*

Table 2. Precision, Recall, F-measure Comparison with Existing Approaches

Dataset	Precision				Recall				F-measure			
	MRegFC	MOCC	ITCC	FCR	MRegFC	MOCC	ITCC	FCR	MRegFC	MOCC	ITCC	FCR
Mv1-E1	**0.75**	0.60	0.63	**0.75**	0.65	**0.67**	0.19	0.61	**0.69**	0.63	0.29	0.67
Mv1-E2	**0.76**	0.62	0.65	0.75	**0.71**	0.65	0.13	0.61	**0.73**	0.63	0.22	0.67
Mv2-E1	**0.70**	0.46	0.54	0.69	**0.64**	0.62	0.23	0.57	**0.66**	0.53	0.32	0.63
Mv2-E2	**0.70**	0.48	0.57	0.69	**0.69**	0.58	0.16	0.56	**0.69**	0.52	0.25	0.63

Table 3. Precision, Recall and F-measure Comparison with Individual Regularizers. (E: Entropy, GI: Gini Index, JE: Joint Entropy)

Dataset	Precision				Recall				F-measure			
	MRegFC	E	GI	JE	MRegFC	E	GI	JE	MRegFC	E	GI	JE
Reuters (21578)	**0.548**	0.409	0.31	0.43	**0.63**	0.546	0.555	0.477	**0.6**	0.46	0.49	0.38
20News Groups	**0.516**	0.3	0.304	0.3	**0.825**	0.767	0.546	0.609	**0.66**	0.43	0.39	0.4
Mv1	**0.756**	0.701	0.689	0.711	**0.653**	0.562	0.554	0.216	**0.7**	0.62	0.61	0.35
Mv2	**0.718**	0.684	0.69	0.702	**0.64**	0.515	0.558	0.268	**0.67**	0.59	0.61	0.39

has an average F-measure of 0.69, while the closest competitor *FCR* achieves a value of 0.65. This clearly demonstrates that *MRegFC* yields consistently better quality clusters compared to the existing algorithms.

Fig. 1 presents the variation of F-measure (using the *Mv2* dataset), as the number of clusters and row-clusters increases from 2 to 10, for *MRegFC* and other algorithms (*MOCC*, *FCR* and *ITCC*). This result was used to choose the number of co-clusters in the algorithm. It can be seen that as the number of clusters increases beyond 5, *MRegFC* consistently outperforms the other algorithms by a convincing margin.

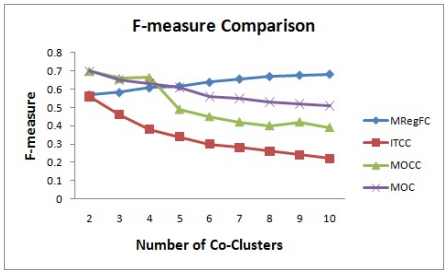


Fig. 1. F-measure vs no. of co-clusters (Mv2)

3.2 Comparison with Individual Regularizers

To quantify the benefit of incorporating multiple regularizers, we also compared *MRegFC* with similar algorithms that include only one of the *Entropy*, *Gini Index*, and *Joint Entropy* regularizers. Table 3 presents the comparison results on the different datasets in terms of F-measure. Clearly, *MRegFC* outperforms the techniques using individual regularizers. Since *MRegFC* also achieves the lowest RMSE of all techniques across all data sets (Table 4, we conclude that the need for incorporating multiple regularizers, as in *MRegFC*, cannot be overemphasized.

We also varied the parameter T_{uv} over a large range on all the datasets and observed that training time and RMSE do not vary much with change in T_{uv}. This demonstrates the robustness of the proposed approach with respect to the input parameter T_{uv}. We omit the details due to space constraints.

Table 4. RMSE comparison

Dataset	MRegFC	Entropy	Gini Index	Joint Entropy	FCR
Reuters (21578)	**1.37**	1.47	1.51	1.58	1.4
20News Groups	**1.45**	1.56	1.56	1.57	1.56
Mv1	**1.36**	1.39	1.45	1.56	1.48
Mv2	**1.23**	1.28	1.3	1.51	1.4
Jester-1	**17.68**	20.47	20.68	22.12	20.64
Jester-2	**17.55**	22.55	20.56	25.7	20.02
Classic3 (CRAN)	**1.04**	1.09	1.13	1.16	1.46
Classic3 (MED)	**1.27**	1.57	1.59	1.59	1.59

4 Conclusion

We present a novel fuzzy co-clustering framework that simultaneously incorporates multiple regularizers namely Entropy, Gini Index, and Joint Entropy while trying to maximize the degree of aggregation. The approach can handle categorical and numerical data in addition to the highly sparse high dimensional data, without over-fitting. Furthermore, unlike existing algorithms, we provide an appropriate range of values for tuning the various parameters to obtain high quality results. We demonstrate superior performance, in terms of several quality measures such as precision, recall, F-measure and RMSE compared to the prior approaches as well as algorithms using individual regularizers.

References

[1] Madiera, S., Oliveira, A.: Biclustering algorithms for biological data analysis: A survey. IEEE/ACM Trans. Computational Biology and Bioinformatics 1, 24–45 (2004)

[2] Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2001, pp. 269–274. ACM, New York (2001)

[3] Dhillon, I.S., Mallela, S., Modha, D.: Information theoretic co-clustering. In: Proceedings of the Ninth ACM SigKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 89–98. ACM Press, New York (2003)

[4] George, T., Merugu, S.: A scalable collaborative filtering framework based on co-clustering. In: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM 2005, pp. 625–628. IEEE Computer Society, Washington, DC (2005)

[5] Banerjee, A., Dhillon, I., Ghosh, J., Merugu, S., Modha, D.S.: A generalized maximum entropy approach to bregman co-clustering and matrix approximation. In: Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2004, pp. 509–514. ACM, New York (2004)
[6] Sahami, M., Hearst, M., Saund, E.: Applying the multiple cause mixture model to text categorization. In: International Conference on Machine Learning (1996)
[7] Tjhi, W.C., Chen, L.: A partitioning based algorithm to fuzzy co-cluster documents and words. Pattern Recognition Letters 27, 151–159 (2006)
[8] Miyamoto, S., Mukaidono, M.: Fuzzy c-means as a regularization and maximum entropy approach. In: Proceedings of IFSA, vol. 2, pp. 86–92 (1997)
[9] Oh, C.H., Honda, K., Ichihashi, H.: Fuzzy clustering of categorical multi-variate data. In: Proceedings of IFSA/NAFIPS, Vancouver, USA, pp. 2154–2159 (2001)
[10] Kummamuru, K., Dhawale, A., Krishnapuram, R.: Fuzzy co-clustering of documents and keywords. In: IEEE International Conference on Fuzzy Systems (2003)
[11] Frigui, H., Nasraoui, O.: Simultaneous clustering and attribute discrimination. In: Proceedings of FUZZIEEE, pp. 158–163 (2000)
[12] Frigui, H., Nasraoui, O.: Simultaneous categorization of text documents and identification of cluster-dependent keywords. In: Proceedings of FUZZIEEE, pp. 158–163 (2001)
[13] Shafiei, M.M., Milios, E.E.: Model based overlapping co-clustering. In: SDM, Maryland, USA (2006)
[14] MacKay, D.: Information theory, inference, and learning algorithms. Cambridge University Press (2003)
[15] Dumitrescu, D., Lazzerini, B., Jain, L.: Fuzzy sets and their applications to clustering and training. CRC Press, Boca Raton (2000)
[16] Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval 4, 133–151 (2001)

Dynamic Ensemble of Ensembles in Nonstationary Environments

Xu-Cheng Yin[1], Kaizhu Huang[2], and Hong-Wei Hao[3]

[1] School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China
[2] Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China
[3] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
xuchengyin@ustb.edu.cn, kaizhu.huang@xjtlu.edu.cn, hongwei.hao@ia.ac.cn

Abstract. Classifier ensemble is an active topic for learning from non-stationary data. In particular, batch growing ensemble methods present one important direction for dealing with concept drift involved in non-stationary data. However, current batch growing ensemble methods combine all the available component classifiers only, each trained independently from a batch of non-stationary data. They simply discard interim ensembles and hence may lose useful information obtained from the fine-tuned interim ensembles. Distinctively, we introduce a comprehensive hierarchical approach called Dynamic Ensemble of Ensembles (DE^2). The novel method combines classifiers as an ensemble of all the interim ensembles dynamically from consecutive batches of non-stationary data. DE^2 includes two key stages: (1) Component classifiers and interim ensembles are dynamically trained; (2) the final ensemble is then learned by exponentially-weighted averaging with available experts, i.e., interim ensembles. We engage Sparsity Learning to choose component classifiers selectively and intelligently. We also incorporate the techniques of Dynamic Weighted Majority, and Learn^{++}.NSE for better integrating different classifiers dynamically. We perform experiments with the data in a typical non-stationary environment, the Pascal Large Scale Learning Challenge 2008 Webspam Data, and compare our DE^2 method to other conventional competitive ensemble methods. Experimental results confirm that our approach consistently leads to better performance and has promising generalization ability for learning in non-stationary environments.

Keywords: Ensemble of ensembles, classifier ensemble, growing ensemble, sparsity learning, nonstationary environment, concept drift.

1 Introduction

Many real-world applications for data learning and mining appear in nonstationary environments, where the underlying data distribution changes over time (i.e., concept drift), such as climate or financial data analysis, network intrusion, spam and fraud detection, information retrieval, and web mining. Informally, concept drift refers to a change in the class (concept) definitions over time, i.e., a set of examples has legitimate class

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 76–83, 2013.

labels at one time and has different legitimate labels at another time. An environment from which such data is obtained is a nonstationary environment. Concepts may change suddenly or gradually, and if we view concepts as shapes in a representation space, then they can change their shape, size, and location ([1]).

Recently, learning such nonstationary data with concept drift has started to received much attention and be an active topic in machine learning and data mining. Methods designed for concept drift can be characterized in several different ways, such as single classifier versus ensemble-based approaches, or online versus batch algorithms. Single classifier based approaches track concept drift, train, and update one classifier incrementally in a nonstationary environment, e.g. STAGGER ([2]) and FLORA ([3]). More popularly, classifier ensemble has been intensively studied to be an effective technique for overcoming the limitations of individual classifiers' accuracy and stability. Consequently, a variety of ensemble based approaches have been investigated and proposed for learning nonstationary data with concept drift ([4]). Generally, there are mainly two categories: the fixed ensemble and the growing ensemble. The fixed ensemble has fixed component learners trained in advance, e.g., Weighted Majority ([5]), Warmuth et al.'s methods ([6,7]). Given new data, it only updates the combination rules or weights of the ensemble, or updates the parameters of existing ensemble components with online learning. Contrarily, the growing ensemble dynamically adds or/and removes components and updates voting weights with each incoming dataset. More specifically, **online growing ensembles**, e.g. Dynamic Weighted Majority (DWM) ([1,8]) and AddExp ([9]), utilize online (incremental) algorithms; however **batch growing ensembles** use batch algorithms, i.e., repeated applications of off-line learning algorithms to process batches of training examples. More researchers are interested on batch growing ensemble technologies, e.g., Streaming Ensemble Algorithm (SEA) ([10], StreamMiner [11]), Selectively Recursive Ensemble Approach (SREA and REA) ([12,13]), and Learn^{++}.NSE ([14]).

In this paper, following the batch growing ensemble, we propose a more adaptive method, Dynamic Ensemble of Ensembles (DE^2), for learning nonstationary data. DE^2 combines classifiers dynamically from consecutive batches of nonstationary data. Firstly, we train interim ensembles using general batch growing ensemble approaches to deal with concept changes over time. At each time (when a batch of data come), one interim ensemble is constructed by weighted majority voting. We engage the Sparsity Learning in order to learn the weight of each component classifiers selectively and intelligently. The Dynamic Weighted Majority (DWM) and Learn^{++}.NSE can be also incorporated into this step. Then, we manage to learn an ensemble for combining all available interim ensembles. The final ensemble of DE^2 is weighted-majority voted by all past and current interim ensembles. The proposed framework is justified theoretically with a shifting regret bound analysis and also validated extensively on real-world benchmark datasets. While current batch growing ensemble methods resort to combining component classifiers only, our proposed DE^2 attempts to combine all interim ensembles. In other words, existing batch growing ensemble methods usually discard previous interim ensembles when a new batch of data come. In comparison, the novel DE^2 aims to take full advantages of knowledge from these previous interim "experts".

The rest of this paper is organized as follows. Section III demonstrates our Dynamic Ensemble of Ensembles for learning from nonstationary environments. Several comparative experiments with the Pascal Competition 2008 spam data are demonstrated in Section IV. Finally, conclusions are presented in Section V.

2 Learning Data with Dynamic Ensemble of Ensembles

The stream data in a nonstationary environment is separated to chunks, i.e., a series of training datsets $D^t = x^t(i) \in X, y^t(i) \in Y, i = 1, \ldots, m^t$, where t is a time index. For simplicity, we assume the classification task is a two-class problem, and specifically set $y^t \in \{-1, 1\}$. $x^t(i)$ is the i^{th} instance of the dataset, drawn from an unknown distribution $P^t(x, y)$ in the environment at time t. At time $t + 1$, a new batch of data drawn from $P^{t+1}(x, y)$ is arrived.

2.1 Dynamic Ensemble of Ensembles (DE^2)

DE^2 is a growing ensemble with batch learning, which includes two stages. In the first stage, at each time an interim ensemble is constructed by weighted majority voting. In the second stage, a final ensemble is also weighted-majority voted by all former and current interim ensembles as a global problem. Weights of classifiers in one interim ensemble can be learned with sparsity learning, and weights of components in the final ensemble are set with exponentially-weighted averaging.

Interim Ensemble with Sparsity Learning In most cases, ensemble of some parts of available component classifiers is better than ensemble as a whole. This leads to sparsity learning for the combination of multiple classifiers. A sparse model representation can improve the generalization performance and computational efficiency. Consequently, we use sparsity learning to construct the interim ensemble in our DE^2 method.

As described above, given one classifier with each time, at time t, we will get t classifier components, $h_1, h_2, \ldots, h_t$. Using weighted majority voting, a combined classifier for a given instance x at time t is,

$$H_t(x) = (w^t)^T h^t(x) \tag{1}$$

where $h^t(x) = [h_1(x) \ldots h_t(x)]^T$.

At time t, given the training data $D^t = x^t(i) \in X, y^t(i) \in Y, i = 1, \ldots, m^t$, and using the sparsity learning with the least squares loss, the optimization for the weights, w^t, is

$$\min_{w^t} \sum_{i=1}^{m^t} \frac{1}{2}((w^t)^T h^t(x^t(i)) - y^t(i))^2 + \lambda \parallel w^t \parallel_1 \qquad \text{s.t.} \quad w^t \geq \mathbf{0} \tag{2}$$

where λ is the control parameter for the sparsity regularization. With the empirical logistic loss, the optimization is converted to

$$\min_{w^t} \sum_{i=1}^{m^t} \log(1 + e^{-y^t(i)((w^t)^T h^t(x^t(i)))} + \lambda \parallel w^t \parallel_1 \qquad \text{s.t.} \quad w^t \geq \mathbf{0} \tag{3}$$

Lastly, we normalize the weights w^t with

$$w^t = w^t / \parallel w^t \parallel_1 \tag{4}$$

Note that the first stage of DE^2 with interim ensembles can also be performed by many other batch growing ensemble approaches, e.g., Dynamic weighted majority ([1]), Learn^{++}.NSE ([14]).

Final Ensemble with Dynamic Weighted. Currently, most batch growing ensemble methods resort to combining component classifiers only, i.e., usually discard previous interim ensembles when a new batch of data come. More rationally, taking full advantages of knowledge from previous interim "experts" has a more attractive profit. As a result, our novel DE^2 method combines classifiers as an ensemble of all the interim ensembles dynamically from consecutive batches of nonstationary data.

After the weighted majority ensemble with sparsity learning at time t, we will get an interim ensemble H_t in (1). Consequently, all the former and current interim ensembles are,

$$\{H_1, H_2, \ldots, H_t\} \tag{5}$$

Exponentially-weighted averaging is a popular method for ensemble with online learning ([5,6,15]). We use exponentially-weighted averaging to combine interim ensembles.

Given t interim ensembles at time t in (5), initial positive weight $\phi_{1,1} = 1$ for the starting classifier h_1, or the interim ensemble H_1 , and a learning rate $\eta > 0$, the final ensemble at time t is constructed by a weighted average,

$$H^t(x) = \frac{\sum_{t_1=1}^{t_1=t} \phi_{t,t_1} H_{t_1}(x)}{\sum_{t_1=1}^{t_1=t} \phi_{t_1,t}} \tag{6}$$

The general exponentially-weighted averaging updates the weights by

$$\phi_{t,t_1} = \phi_{t-1,t_1} e^{-\eta \ell(H_{t_1})} \tag{7}$$

where $\ell(H_{t_1})$ is the loss of the interim ensemble H_{t_1}, e.g., the classification error rate (the $1-0$ loss function) or the least squares loss.

In our DE^2, we use an another way for updating weights, which is more suitable for ensemble with classifier sequences. The ensemble with this strategy is called as the fixed shares forecaster (ensemble) ([6,7,15]), which is equivalent to the exponentially weighted average forecaster (ensemble) ([15]). The weight updating equations are

$$\phi_{t,t_1} = (1-\alpha)\varphi_{t,t_1} + \alpha \frac{\sum_{j\neq t_1}^{j=t} \varphi_{t,j}}{t} \tag{8}$$

and

$$\varphi_{t,t_1} = \phi_{t-1,t_1} e^{-\eta \ell(H_{t_1})} \tag{9}$$

where $\alpha \in [0,1]$ is another control setting.

INPUT:

Training set $\{D_1, D_2, D_3, \dots\}$, Learning rate η, control parameter α, initial weight ϕ_0.

INITIALIZATION:

$H = \Phi, h = \Phi, \phi_{1,0} = \phi_0$.

for t = 1,2,3,...

STEP 1: Train interim ensembles

$\{H_t, h_t\}$ = middle_ensemble_train(D_t, H, h); $h = h \bigcup h_t$; $H = H \bigcup H_t$;

STEP 2: Train the final ensemble

for $t_1 = 1, 2, \dots, t$

$$\varphi_{t,t_1} = \phi_{t-1,t_1} e^{-\eta \ell(H_{t_1})}; \qquad \phi_{t,t_1} = (1-\alpha)\varphi_{t,t_1} + \alpha \frac{\sum_{j \neq t_1}^{j=t} \varphi_{t,j}}{t};$$

$$H^t(x) = \frac{\sum_{t_1=1}^{t_1=t} \phi_{t,t_1} H_{t_1}(x)}{\sum_{t_1=1}^{t_1=t} \phi_{t,t_1}};$$

OUTPUT:

The final ensembles $\{H^1, H^2, H^3, \dots\}$.

Fig. 1. Algorithm for the DE^2 learning (**Algorithm I**)

Here, DE^2 combines classifiers dynamically from consecutive batches of nonstationary data. In the first stage, interim ensembles are dynamically learned with batch growing ensembles, e.g., Sparsity learning, Dynamic weighted majority, or Learn^{++}.NSE. In the second stage, a final ensemble is combined with all past and current interim ensembles according to their series performances. We use exponentially weighting and dynamically updating strategy with an online style for these interim ensembles. As a result, we call our proposed method as Dynamic Ensemble of Ensembles (DE^2).

2.2 Learning Algorithm

As demonstrated above, at time t, our learning algorithm includes two stages which is shown in Figure 1: Firstly, train component classifiers and construct interim ensembles (*middle_ensemble_train()*); secondly, learn the final ensemble with exponentially updating of weights ((6) $\sim$ (8)). In Figure 1, we use the classification error rate as the loss of the interim ensemble, i.e., $\ell(H_t)$ is equal to the classification error rate of H_t.

In the first stage of DE^2, we can use three ensemble approaches, i.e., Sparsity Learning Combination (1), Dynamic weighted majority ([1]) (if the base classifier is an incremental learner), Learn^{++}.NSE ([14]), to train component classifiers and construct the interim ensemble H_t in Figure 1. When training interim ensembles with Sparsity Learning. In our implementation, we use the least squares optimization (2) for computing the sparse weights.

Our method can also use Dynamic Weighted Majority (DWM) ([1]) to construct the interim ensemble (*middle_ensemble_train()* in Figure 1), where the base classifier must be an incremental learner. In our DE^2 with DWM, the interim ensemble at each time step is the DWM, which incrementally creates, removes, and trains base classifiers. Moreover, with Learn^{++}.NSE ([14]) for *middle_ensemble_train()*, at each time step, the Learn^{++}.NSE adds a new classifier at each time, dynamically updates classifiers' weights, and weightedly combines component classifiers.

3 Experiments

In our experiments, one new classifier is trained at one time, and at the time t, there are t classifiers $\{h_1, h_2, \ldots, h_{t-1}, h_t\}$. At the time step t, the SINGLE approach uses only one new classifier h_t to classify testing samples; the AVERAGING method combines all t available classifiers with averaging; and the SPARSITY approach ensembles these t classifiers with sparsity learning.Moreover, we also compare two popular growing ensemble methods for nonstationary data learning with concept drift: Dynamic Weighted Majority (DWM) ([1]), and Learn^{++}.NSE ([14]). The base learner of DWM must be an **incremental** classifier.

In our experiments, three kinds of interim ensembles are utilized: DWM, Learn^{++}. NSE, and SPARSITY. Two kinds of base learners are used: Supported Vector Machines (SVM) and Naive Bayes Classifiers (NBC). Because the DWM needs an incremental base learner, Online NBC is used in DWM. Consequently, for the SVM base learner, our experimental DE^2 algorithms are DE^2_SPARSITY (*middle_ensemble_train()* using SPARSITY in Figure 1), and DE^2_NSE (using Learn^{++}.NSE). For the NBC base learner, our experimental DE^2 algorithms are DE^2_SPARSITY (using SPARSITY), DE^2_DWM (using DWM with the incremental naive Bayes classifier), and DE^2_NSE (using Learn^{++}.NSE).

In these experiments, for each classifier with each data set, the classification system is performed for ten times. And the presented results are averaged on all ten times.

3.1 Experiments with Pascal 2008 Webspam Data

The Pascal Large Scale Learning Challenge 2008 webspam data set is the Web spam corpus, which consists of $350,000$ Web spam pages ([16]). All positive examples are taken and the negative examples are created by randomly traversing the Internet starting at well known (e.g. news) web-sites. As we know, the Internet is a typical nonstationary environment. Consequently, classifying web spam data is a difficult problem with a heavy concept drift. We preprocess this dataset by the way in LIBSVM ([17]), where they treated continuous n bytes as a word: trigram if $n = 3$ and unigram if $n = 1$. They used word count as the feature value and normalize each instance to unit length. In this experiment, we use the unigram features the number of which is 254. Moreover, the features are performed with dimension reduction to 30 dimensions by principal components analysis (PCA). This set includes $350,000$ samples. At each time step, the former 875 samples are used for training, and the next 875 samples are for testing.

Table 1. The average classification performance comparisons with SINGLE, ensembles, and DE^2 methods using SVM and NBC base classifier

SVM	SINGLE	AVERAGING	SPARSITY	Learn^{++}.NSE	DE^2_NSE	DE^2_SPARSITY		
Accuracy(%)	91.8 ± 0.2	92.5 ± 0.2	92.1 ± 0.3	92.5 ± 0.2	92.6 ± 0.1	**92.7** ± 0.3		
Sparsity	1.0	100.5	9.5	38.9	47.8	19.0		
NBC	SINGLE	AVERAGING	SPARSITY	Learn^{++}.NSE	DE^2_NSE	DE^2_SPARSITY	DWM	DE^2_DWM
Accuracy(%)	81.9 ± 0.0	64.5 ± 0.0	82.1 ± 0.1	82.3 ± 0.1	**82.4** ± 0.1	82.2 ± 0.1	80.8 ± 0.2	80.9 ± 0.1
Sparsity	1.0	100.5	12.0	19.3	21.0	17.0	11.1	16.4

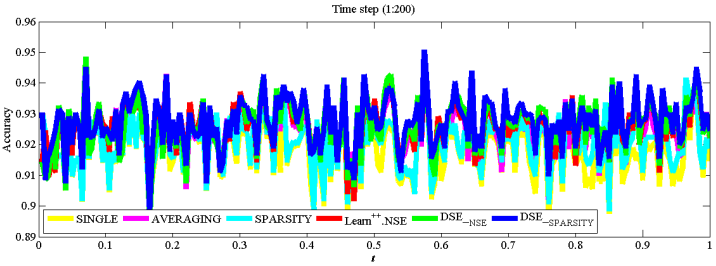


Fig. 2. Comparative performances on the Webspam dataset with the RBF SVM base learner

The results with the SVM base learner are shown in Figure 2 and Table 3.1, where the LIBSVM is used with the RBF kernel. The parameters of LIBSVM are decided by the grid search procedure. Some remarks are highlighted from these experimental results. Firstly, because there are enough samples at each time (875 samples), the SINGLE method shows a good performance. Moreover, the AVERAGING approach has a quite high classification accuracy (92.5% in Table 3.1). Secondly, the growing ensembles (Learn^{++}.NSE, DE^2_NSE and DE^2_SPARSITY) show a much better performance. In particular, our DE^2 methods have more impressive results. Specifically, the DE^2_SPARSITY has the highest accuracy (92.7% in Table 3.1) among all experimental methods. The results with the Naive Bayes Classifier (NBC) base learner are similar to the ones with SVM.

As described in Table 3.1, some remarks are as follows. Firstly, the growing ensemble techniques (DWM, Learn^{++}.NSE and DE^2) are better than the SINGLE method, and simple ensemble methods (AVERAGE and SPARSITY). This shows that the growing ensemble approach is an effective technology for learning nonstationary data. Secondly, our DE^2 approaches, DE^2_DWM, DE^2_NSE and DE^2_SPARSITY, are better than their interim ensembles, DWM, NSE and SPARSE, respectively. Specifically, our proposed DE^2 approach, including DE^2_SPARSITY and DE^2_NSE, all are competitive to and even better than famous growing ensembles (DWM and Learn^{++}.NSE). Very interestingly, with the SVM base learner, the DE^2_SPARSITY has the best performance among all experimental methods; with the NBC base learner, our DE^2_NSE is the best one. Finally, similar to DWM and Learn^{++}.NSE, our DE^2 method has a rather low sparsity (only using a few trained classifiers) without requiring access to previously seen data. All these experimental results confirm that our approach has a promising generalization performance for learning in nonstationary environments.

4 Conclusions

Classifier ensemble is widely considered to be an effective technique for improving the accuracy and stability for a nonstationary data classification system. In this paper, we introduce a classifier ensemble approach for incremental learning of concept drift. This growing ensemble method, Dynamic Ensemble of Ensembles (DE^2), combines classifiers dynamically from consecutive batches of nonstationary data with two key stages. In the first stage, at each time an interim ensemble is constructed by weighted

majority voting, e.g., SPARSITY learning, Dynamic Weighted Majority (DWM), or Learn^{++}.NSE. In the second stage, a final ensemble is also weighted-majority voted by all former and current interim ensembles as a global problem, and weights of components are set with exponentially-weighted averaging. Experimental results confirm that our method has a very promising generalization performance for learning in nonstationary environments.

References

1. Kolter, J., Maloof, M.: Dynamic weighted majority: An ensemble method for dirfting concepts. Journal of Machine Learning Research 8(2), 2755–2790 (2007)
2. Schlimmer, J., Granger, R.: Beyond incremental processing: Tracking concept drift. In: Proceedings of National Conference on Artificial Intelligence, pp. 502–507 (1986)
3. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning 23(1), 69–101 (1996)
4. Kuncheva, L.: Classifier ensembles for changing environments. In: Proceedings of International Workshop on Multiple Classifier Systems, pp. 1–15 (2004)
5. Littlestone, N., Warmuth, M.: The weighted majority algorithm. Information Computation 108(2), 212–261 (1994)
6. Herbster, M., Warmuth, M.: Tracking the best expert. Machine Learning 32(2), 151–178 (1998)
7. Bousquet, O., Warmuth, M.: Tracking a small set of experts by mixing past posteriors. Journal of Machine Learning Research 3(1), 363–396 (2002)
8. Kolter, J., Maloof, M.: Using additive expert ensembles to cope with concept drift. In: Proceedings of International Conference on Machine Learning, pp. 449–456 (2005)
9. Kolter, J., Maloof, M.: Dynamic weighted majority: An ensemble method for dirfting concepts. In: Proceedings of IEEE International Conference on Data Mining, pp. 123–130 (2003)
10. Street, W., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 377–382 (2001)
11. Fan, W.: Streamminer: A classifier ensemble-based engine to mine concept-drifting data streams. In: Proceedings of International Conference on Very Large Data Bases, pp. 1257–1260 (2004)
12. Chen, S., He, H.: SERA: Selectively recursive approach towards nonstationary imbalanced stream data mining. In: International Joint Conference on Neural Networks, pp. 522–529 (2009)
13. Chen, S., He, H.: Toward incremental learning of nonstationary imbalanced data stream: A multiple selectively recursive approach. Evolving Systems 2(1), 30–50 (2011)
14. Elwell, R., Polikar, R.: Incremental learning of concept drift in nonstationary environments. IEEE Trans. Neural Networks 22(10), 1517–1531 (2011)
15. Shalizi, C., Jacobs, A., Klinkner, K., Clauset, A.: Adapting to non-stationarity with growing expert ensembles, arXiv:1103.09049v2 (2011)
16. Webb, S., Caverlee, J., Pu, C.: Introducing the webb spam corpus: using email spam to identify web spam automatically. In: Proceedings of Third Conference on Email and Anti-Spam (2006)
17. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intelligent Systems and Technology 2(3), 1–27 (2011)

Improvement of the Relaxation Procedure in Concurrent Q-Learning

Kazunori Murakami[1] and Tomoko Ozeki[2]

[1] Graduate School of Engineering, Tokai University, Hiratsuka, Japan
2bdrm021@mail.tokai-u.jp
[2] Department of Human and Information Science, Tokai University, Hiratsuka, Japan
tozeki@tokai.ac.jp

Abstract. In this paper, we point out problems in concurrent Q-learning (CQL), which is one of the adaptation techniques to dynamic environment in reinforcement learning and propose the modification of the relaxation procedure in CQL. We apply the proposed algorithm to the problem of maze in reinforcement learning and validate what kind of behavior the original CQL and the proposed algorithm show for the changes of environment such as the change of goals and the emergence of obstacles.

Keywords: Reinforcement Learning, Dynamic Environment, Concurrent Q-Learning, Relaxation.

1 Introduction

In reinforcement learning [1], which is one of machine learning techniques, the agents interact with the environment by repeated trail-and-error in order to obtain the sequence of appropriate actions to achieve the goals. It is different from supervised learning in a sense that there is no teacher to tell the correct actions and that the agent learns the optimal action based on the rewards which are given by the environment. Since the agents learn the optimal action from the environment, it is said that it is able to achieve its optimal behavior automatically even if the environment changes. However, learning in the past interferes with adaptation to the new environment. In this case, it is necessary to adjust learning parameters. It is still an open problem how to adjust parameters when the environment suddenly changes.

On the other hand, Morris water maze experiment [2] reveals that the rats immediately adapt to the changes of the environment. When the platform is moved to the new location, rats trained to find the hidden platform in the water can directly go to the new one once the new platform is found.

Foster et al. performed simulations of water maze task by combination of TD learning and spatial coordinate learning and obtained the same results as the biological rats [3]. Acquiring the spatial coordinates enables the agents to obtain the shortest path between the current location and the goal and to cope with the environmental changes immediately. Although the algorithm is efficient for

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 84–91, 2013.

the changes of goals, it does not work for the changes such as abrupt emergence of obstacles in the path.

Kaelbling proposed DG learning which learns Q-values for the multiple goals simultaneously and demonstrated that it can adapt the change of the goal [4]. Ollington and Vamplew introduced the eligibility trace in the similar learning to the DG learning and proposed concurrent Q-learning (CQL) which adapts to the change of goals and emergence of obstacles [5].

In this paper, we point out the serious problems in CQL and aim at the improvement in the relaxation procedure. The propose method is applied to the maze problem where the goals and the obstacles are changed. We make comparison between the original CQL and the proposed method.

This paper is organized as follows. The belief introductions for the Q-learning are given in Sec. 2. We describe the eligibility trace in Sec. 3 and concurrent Q-learning in Sec. 4. In Sec. 5, we point out the problems in CQL and propose the improvement in the relaxation procedure. In Sec. 6, we show the results of computer simulations, comparing the proposed method to the original CQL.

2 Q-Learning

Q-learning, which is based on the Q-value, is one of the reinforcement learning algorithms. The Q-value, $Q(s, a)$ represents the cumulative rewards expected to get by taking the action a in the state s. The Q-value is updated during learning by

$$\delta \leftarrow r + \gamma \max_{a' \in A(s')} Q(s', a') - Q(s, a) \tag{1}$$

$$Q(s, a) \leftarrow Q(s, a) + \alpha\delta \tag{2}$$

where r, s, a, s', a' denote the reward, the current state, the current action, the next state, and one of the next possible actions. $A(s')$ is a set of actions which the agent can take in the next state s. α is a learning rate, γ is a discount rate and δ is the TD error. The learning rate and the discount rate are metaparameters. The learning rate controls the speed and the stability of the learning and the discount rate represents the credibility of the past and the future learning.

3 Eligibility Trace

The eligibility trace is a way of giving the appropriate rewards to the state-action pairs which the agent took in the past. Here, Watkins $Q(\lambda)$ is used for updating the eligibility trace [1]. $e(s, a)$ is the eligibility trace for a state-action pair and is set to zero initially. When the action a in a state s is selected, the eligibiliry is set to

$$e(s, a) \leftarrow 1 \ . \tag{3}$$

Then for all the state-action pairs, the following updating

$$Q(s, a) \leftarrow Q(s, a) + \alpha\delta e(s, a) \tag{4}$$

$$e(s, a) \leftarrow \gamma\lambda e(s, a) \tag{5}$$

is executed, where δ is the TD error and λ is the eligibility trace decay factor.

4 Concurrent Q-Learning [4]

Ollington and Vamplew developed the idea of DG learning by Kaelbling and proposed the CQL which can solve the navigation problem for all of the possible goals concurrently. We briefly introduce the CQL-e, one of the CQL techniques.

4.1 Relaxation

When the following equation,

$$\overline{AC} \leq \overline{AB} + \overline{BC} \tag{6}$$

for the distances between three locations A, B, and C in maze holds, this expression can be replaced in the expression of the Q-values as

$$Q^C(s_A, a) \geq Q^B(s_A, a) \times \max_{a' \in A(s_B)} Q^C(s_B, a') \tag{7}$$

where $Q^*(s, a)$ is the Q-value of state s and action a for the goal $*$. This rule is applied to the Q-values for all of the state-action pairs (s^0, a^0) in learning. When this rule does not hold, namely δ^0

$$\delta^0 \leftarrow Q^s(s^0, a^0)\,[r^* + \gamma \max_a Q^*(s', a)] - Q^*(s^0, a^0) \tag{8}$$

is larger than zero, the Q-value $Q^*(s^0, a^0)$ is modified by

$$Q^*(s^0, a^0) \leftarrow Q^*(s^0, a^0) + \alpha\delta^0 \ . \tag{9}$$

The algorithm of CQL is given in Fig. 1. The action selection is performed by the ϵ-greedy method.

5 Problems and Improvements

5.1 Conditions of Simulations

We have applied the CQL algorithm to the maze problem of the grid world to investigate the efficiency of CQL. We consider a grid world in which the field composed of 10×10 grid is divided into two rooms with the wall in Fig. 2. The rooms are connected by two doors. At least one door is open and the agent has to go through one of the doors to reach a goal from a start. Each action of the agent is chosen from one of four directions of up, right, down and left.

To simulate the changing environments, we consider four types of changes, (i) the change of goal, (ii) detour, (iii) blocking and (iv) shortcut. In the case of the change of the goal location, both of the doors are open all the time. After the change of the goal, the agent has to take a shortcut through the other door. In the detour experiment, both of the doors are open at the beginning and then the door which the agent learns as a shortcut is closed. The agent has to search for the shortest path through the other door. In blocking, one door is closed at

```
Initialize Q*(s,a) and e*(s,a)=0 for all *(goal), s, a
Initialize s, a
Repeat:
 Take action a, observe s'
 Choose action a' from s'
 For each goal *:
  e*(s,a) = 1
  delta = r* + gamma x max Q*(s', a') - Q*(s, a)
  for all state-action pairs s0, a0:
    if e*(s0, a0) > 0
      Q(lambda) Update Eq. (4)
    else
      Relaxation
      delta calculation Eq. (8)
      if delta > 0
          Eq. (9)
   eligibility trace update Eq. (5)
s = s', a = a'
```

Fig. 1. Algorithm of the original CQL

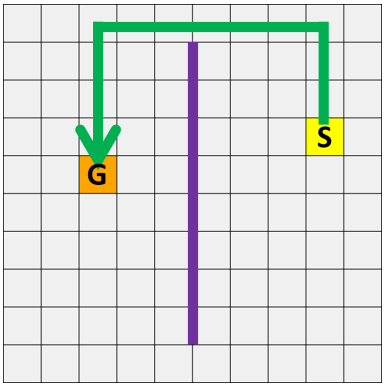


Fig. 2. Maze of the grid world

the beginning and then the door which the agent learns to go through will be closed and the other door will be opened. In the shortcut experiment, one door is closed at first and both of the doors will be opened after the change of the environment. The agent is expected to use the other door to find a shorter path.

In each experiment, we investigate the steps taken by the agent from the start to the goal. The change of the environment occurs after 200 goals, or episodes. One set of the experiment consists of 400 episodes and we perform 100 sets of experiments to take average. The parameters for the experiment are set to 0.1 for the learning rate, 0.95 for the discount rate and 0.1 for the probability of random exploration of ϵ-greedy action selection.

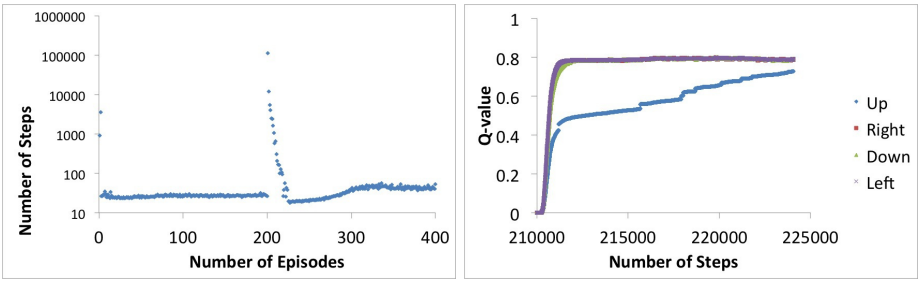


Fig. 3. Performance of the conventional CQL: (Left) Change of number of steps to the goal in the conventional CQL (Right) Time change of Q-values

5.2 Problems in the CQL

Figure 3 (Left) shows the typical result obtained by the simulation of the original CQL. The environmental change, the blocking, occurred after 200 episodes and the figure shows the change of the steps taken by the agent from the start to the goal. It is observable that the agent adapts to the new environment quickly. However, after adapting to the new environment, the number of steps taken is gradually increasing. This means that the agent does not take the shortest path even though it can find it once.

5.3 Relaxation

In order to clarify the cause for the above mentioned problem, we investigate the relaxation procedure in detail. The relaxation procedure is a method where the agent can take shorter path to the goal. For example, when the value of the action going directly from the location A to the location C is smaller then that of the action going from A to C by making a detour through B. Fig. 3 (Right) shows the time development of the Q-values for each actions (up, right, down, left) at a location. All of the Q-values for each action gradually increase and saturate to the same value. Since the relaxation procedure is applied to the Q-values for all of state-action pairs, all of the Q-values are increasing and converge to the high value. This causes inappropriate action selection because the Q-values, which can be a guide for the action selection, are not distinguishable.

5.4 Modified Relaxation Procedure

In order to solve the problem mentioned above, we propose the modified CQL algorithm where not only the Q-value directly going to C but also the Q-value detouring through B is also updated when the shortcut condition (7) is not satisfied. The Q-value detouring through B is reduced as

$$\delta^0 \leftarrow Q^s(s^0, a^0)\,[r^* + \gamma \max_a Q^*(s', a)] - Q^*(s^0, a^0) \tag{10}$$

$$Q^*(s^0, a^0) \leftarrow Q^*(s^0, a^0) + \alpha\delta^0 \tag{11}$$

$$Q^s(s^0, a^0) \leftarrow Q^s(s^0, a^0) \times (1 - \delta^0) \ . \tag{12}$$

This reduction of the Q-value avoids from all saturating Q-values. However, in the proposed algorithm, the values of detours decrease unlimitedly if the Q-values are decreased at each time of calculation for all of the possible goals. When an action is taken in a state, the Q-values $Q^*(s^0, a^0)$ for all the possible locations (s^0, a^0) and all the possible goals $*$ are calculated as in Fig. 1. Without updating the Q-values for detours in each calculation, we preserve the Q-values for detours until the next action is taken and the Q-values are updated just before next action using (i) the maximum of the Q-values and (ii) the minimum of the Q-values. We also performed (iii) the updating at each time for calculations without saving the Q-values. Therefore, we propose three algorithm for decreasing the Q-values for detouring

$$Q^s(s^0, a^0) \leftarrow \max_* \left[Q^s(s^0, a^0) \times (1 - \delta^0)\right] \tag{13}$$

$$Q^s(s^0, a^0) \leftarrow \min_* \left[Q^s(s^0, a^0) \times (1 - \delta^0)\right] \tag{14}$$

$$Q^s(s^0, a^0) \leftarrow Q^s(s^0, a^0) \times (1 - \delta^0) \ . \tag{15}$$

Eq. (13) is an algorithm in which the largest Q-value is used to update the Q-values for detouring. Eq. (14) denotes an algorithm in which the smallest Q-value is used for updating. The third algorithm updates the Q-values using Eq. (15) at the same time as the calculation is performed when the Q-values for detour is larger than the direct path. We call these proposed algorithms as the proposed algorithm (max), the proposed algorithm (min) , and the proposed algorithm (direct).

6 Experiments

6.1 Experimental Results

The experimental results for four types of the environmental changes are given in Fig. 4.

In the experiment of the goal change, the detour and blocking, the proposed algorithms (min, max, and direct) suppress the increase of the steps after the change of the environment. In the original CQL, the steps are increasing after the environmental change. In the blocking experiment, the proposed algorithm (direct) shows the fastest learning convergence of all. In the shortcut experiment, none of the algorithm could notice the environmental changes.

Figure 5 shows how the Q-values of a location evolves after the environmental changes. In the proposed algorithm (max), the Q-values for all the actions are increasing, but still they can be distinguishable. As for the proposed algorithm (min), although the Q-values of actions tend to increase slightly, the action of the highest Q-value is distinguishable compared to the others.

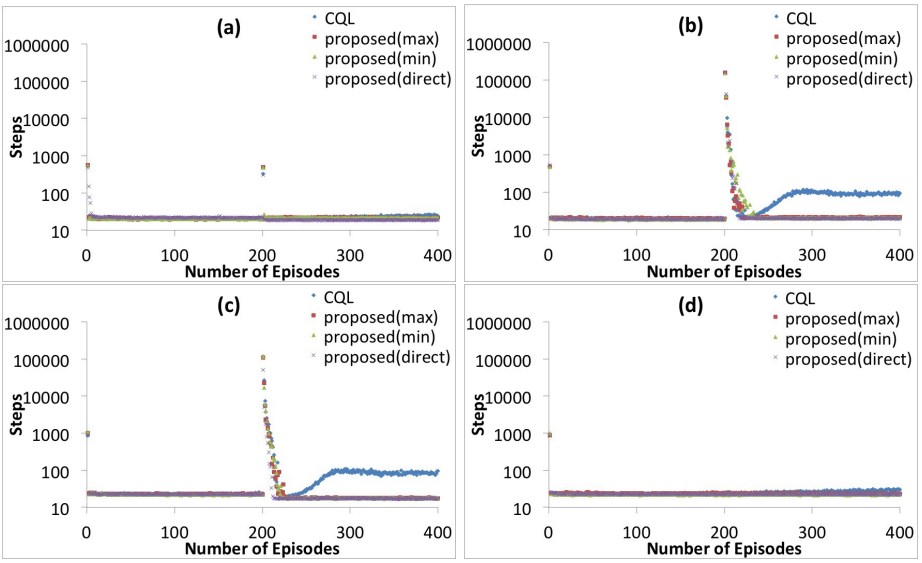


Fig. 4. Changes of time steps: (a) the change of goal (b) detour (c) blocking (d) shortcut

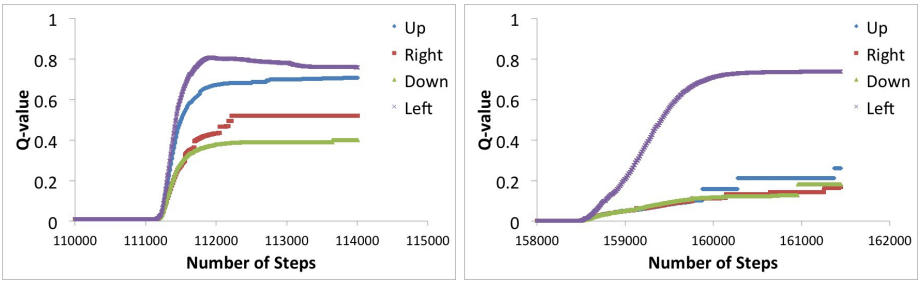


Fig. 5. Time change of the Q-values of a location: (Left) the proposed algorithm (max), (Right) the proposed algorithm (min)

7 Conclusions

In this paper, we have pointed out the problems in CQL and proposed the modified algorithm concerning the relaxation procedure. In CQL, the steps needed to reach the goal increase although the agent can find a shorter path right after the environmental change. This is due to the relaxation procedure which enlarges the Q-values of the direct path and this causes the increase of all the Q-values. We proposed the modified algorithm with the combination of decreasing the Q-values for detours. The proposed algorithm adapts to the new environment at the same speed of the original CQL and improves the problem of increasing time steps after the environmental change. As future problems, we need to investigate

the effects of modification of the algorithm on the adaptation speed to the new environment and the stability of the adaptation in details.

References

1. Sutton, R.S., Barto, A.G.: Reinforcement learning - an introduction. MIT Press (1998)
2. Morris, R.G.M.: Spatial localization does not require the presence of local cues. Learning and Motivation 12, 239–260 (1981)
3. Foster, D.J., Morris, R.G.M., Dayan, P.: A model of hippocampally dependent navigation using the temporal difference learning rule. Hippocampus 10, 1–16 (2000)
4. Kaelbling, L.P.: Learning to achieve goals. In: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (1993)
5. Ollington, R.B., Vamplew, P.W.: Concurrent Q-learning: reinforcement learning for dynamic goals and environments. International Journal of Intelligent Systems 20, 1037–1052 (2005)

Modularity Embedding

Wenye Li

Macao Polytechnic Institute,
Rua de Luís Gonzaga, Macao SAR, China
wyli@ipm.edu.mo
http://staff.ipm.edu.mo/~wyli

Abstract. A number of nonlinear dimensionality reduction, or graph embedding, techniques have been proposed recently. These embedding techniques aim to provide a low-dimensional depiction of graphs while preserving certain properties of the data. In this manuscript we propose a novel graph embedding method which tries to optimize the "modularity" of graphs during dimensionality reduction. The embedding method has a simple formulation and is naturally relaxed and solved by a convex semi-definite program, with the guaranteed global optimum. We evaluate the performance of the method with a variety of examples and the method reports promising results in inspecting the cluster structures of graphs.

Keywords: Dimensionality Reduction, Modularity Embedding, Semi-definite Programming.

1 Introduction

The study of graph embedding has attracted much research interest recently. It often appears as a kind of dimensional reduction techniques for high-dimensional learning problems. With the assumption that the data have an intrinsic dimension that is significantly lower than the number of features they appear to have, the dimensionality reduction techniques have been widely studied during the past decades as a treatment to the "curse of dimensionality". The techniques include principal component analysis (PCA) [1] and metric multidimensional scaling (MDS) [2]. These linear methods project the data from a high-dimensional space into a low-dimensional subspace by either maximizing the projected variance or best preserving the pairwise squared distance among the data.

More recently, there have been a lot of research on the nonlinear methods for dimensionality reduction. These methods build upon but go beyond the classical linear methods. They assume the data are from a low-dimensional manifold embedded in a high-dimensional space, which is more general than the assumption of subspace by linear methods. To get the intrinsic representation of the data, a neighborhood graph is often constructed by connecting each data point to its nearest neighbors. The graph serves as an approximation to the data manifold. With such a graph, people can get the low-dimensional representation of the data by further applying some analysis techniques, such as the spectral graph

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 92–99, 2013.

methods [3]. Some of the recent manifold learning algorithms include Isomap, LLE and others [4–9].

These nonlinear dimensionality reduction techniques provide a natural solution for graph embedding. With these techniques, different signatures of the underlying manifold are preserved, such as the geodesic distances between inputs, the local angles and distances, etc. These properties make the embedding algorithms suitable for different applications.

In this paper, we provide a novel model for graph embedding. Comparing with existing methods, our model is based on a more *semantic* criterion, the "modularity" measure of networks. It tries to find a low-dimensional depiction of networks while preserving the cluster structure inherent in the network. This criterion tries to organize intra-cluster vertices closer, while separating inter-cluster vertices apart. This specific criterion makes the method to give quite different results from other embedding techniques.

To directly compute the embedding results of the model is difficult. We resort to a relaxation technique, and we are able to solve the model by a positive semi-definite program, which provides an effective and efficient solution.

2 Background

Dramatic advances have been made in the development of semi-definite program (SDP) techniques[10, 11] recently. It is concerned with the optimization problems over symmetric positive semi-definite matrix variables with linear cost function and linear constraints. Denote by $\mathbf{S}^n$ the space of all $n \times n$ real symmetric matrices, equipped with the inner product $\langle X, Y \rangle = tr\left(X^T Y\right) = \sum_{i,j=1}^{n} x_{ij} y_{ij}$. A symmetric matrix X is positive semi-definite if all its eigenvalues are nonnegative; we write $X \succeq 0$. SDP deals with optimization problems of the type

$$\min_{X \succeq 0} tr\left(A^T X\right) \tag{1}$$

subject to

$$tr\left(A_i^T X\right) = c_i,\ i = 1, 2, \cdots, m \tag{2}$$

in variables $X \in \mathbf{S}^n$.

SDP has a structure that makes its solution computationally tractable by interior-point methods. It is now used in a host of applications, including relaxation of combinatorial optimizations and learning problems [12–14, 8].

3 Modularity Embedding

3.1 Model

For an undirected network $G = (V, E)$, where $V = \{v_1, \cdots, v_n\}$ is a set of vertices and E is a set of edges connecting vertex pairs in V. Let w_{ij} be an element of the adjacency matrix W of the network, which gives the number of edges between

vertices v_i and v_j. We further denote $m_i = \sum_j w_{ij}$ as the degree of v_i and $m = \frac{1}{2}\sum_i m_i$ as the total edge number.

Assuming the degree m_i associated with each vertex v_i is preserved, under uniform random selection the expected number of edges between two vertices v_i and v_j is $\frac{m_i m_j}{2m}$. A value $b_{ij} = w_{ij} - \frac{m_i m_j}{2m}$ gives the observed number of edges minus the expected number of edges between v_i and v_j. It is positive if the edge weight between v_i and v_j is larger than the expected weight between them. It is zero or negative otherwise. The value quantifies a kind of affinities between the two nodes. A larger b_{ij} means a stronger connection between v_i and v_j and a higher chance that the two nodes are in the same clusters.

An $n \times n$ square matrix B with elements $\{b_{ij}\}$ is called a modularity matrix, which has been found useful in seeking the underlying cluster (community) structure in many networks [15]. It has a nice property. All rows and columns of the matrix sum to zero, i.e.,

$$\sum_j b_{ij} = 0, \ and, \ \sum_i b_{ij} = 0 \tag{3}$$

for all i and j respectively.

Our work wishes to find a d-dimensional (typically $d \ll n$) embedding of the network G such that the cluster structure is preserved. To do this, we propose the following *modularity embedding* model

$$\min_X \sum_{ij} b_{ij} \times \ell\left(x_i, x_j\right) \tag{4}$$

subject to the constraint

$$\sum_i x_i^T x_i = n. \tag{5}$$

Here $X = [x_1, \cdots, x_n]$, where x_i gives the d-dimensional coordinate of vertex v_i and $\ell\left(x_i, x_j\right)$ is the squared Euclidean distance between x_i and x_j.

Using the model, if the value of b_{ij} is positively large, which indicates a high chance of v_i and v_j being in the same cluster, we wish to have a short distance between them in the embedding. Otherwise, if the value is negatively large, which indicates a low possibility of the two vertices being in the same cluster, we then wish to separate them apart as further as possible.

3.2 SDP Relaxation

It is not easy to solve the proposed model directly [16, 17]. But it can be relaxed by a positive semi-definite program. This is done by noting that $\ell\left(x_i, x_j\right) = x_i^T x_i + x_j^T x_j - 2x_i^T x_j$. The optimization objective becomes

$$\sum_{ij} b_{ij} \times \left(x_i^T x_i + x_j^T x_j - 2x_i^T x_j\right) = -2\sum_{ij} b_{ij} x_i^T x_j \tag{6}$$

due to the properties in Eq. (3).

Let $S = X^T X$, which implicitly enforces a constraint on positive semi-definiteness of S, a constraint on the trace $tr(S) = n$ which comes from Eq. (5) and a constraint on the rank $rank(S) = d$. With these constraints, our objective can be written equivalently as the maximization of $tr(BS)$.

The rank constraint makes the problem difficult. A special case of the problem becomes an NP-hard binary partition problem discussed in [18]. Fortunately, there is a simple relaxation that is often found effective in practice. We go on with the optimization by neglecting this inconvenient rank constraint and the problem becomes

$$\max_{S \succeq 0} tr(BS) \tag{7}$$

subject to

$$tr(S) = n. \tag{8}$$

The problem becomes a standard SDP and can be solved in polynomial time.

With matrix S, it is straightforward to recover the embedding results X by matrix decomposition. Assume $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$ be the eigenvalues of S. Let V_i^k denote the i^{th} element of the k^{th} eigenvector V^k, with eigenvalue λ_k. Then each element of the matrix S can be written as

$$s_{ij} = \sum_{k=1}^{n} \lambda_k V_i^k V_j^k. \tag{9}$$

Thus the d-dimensional embedding coordinate of vertex v_i is given by

$$x_i = \left(\sqrt{\lambda_1} V_i^1, \cdots, \sqrt{\lambda_d} V_i^d\right)^T. \tag{10}$$

4 Evaluations

We used both real-world and synthetic datasets to evaluate the performance of the modularity embedding method.

4.1 An Artificial Dataset with Four Clusters

We randomly generated a dataset with four clusters. Each cluster has 50 points drawn from a normal distribution centering around the four corners of a tetrahedron in a 3-dimensional space (see Fig. 1). We constructed a network by connecting each point with its six nearest neighbors and compared 1-dimensional and 2-dimensional embedding results from the Isomap algorithm and modularity embedding.

Fig. 2(a) and 3(a) give the embedding results which tried to preserve the pairwise distance with the Isomap algorithm. From the results we can see, as the intrinsic dimension of the data is 3, when we tried to embed the data in lower dimensions, the cluster structure was not well preserved. In 1-dimensional embedding, three clusters (in black, blue and red respectively) were mixed together

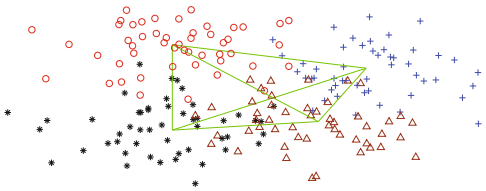
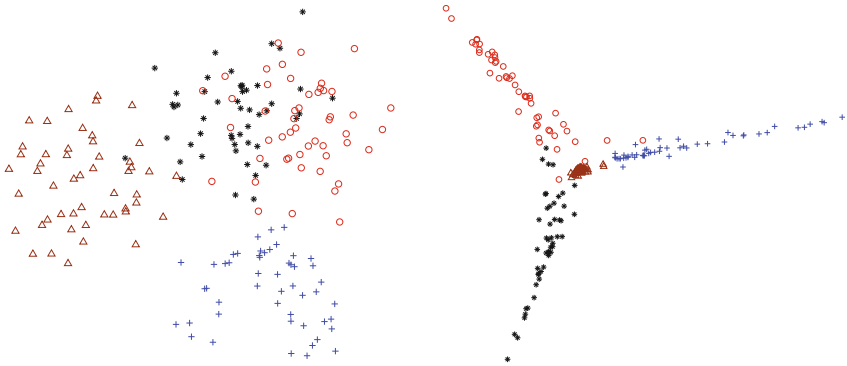


Fig. 1. A synthetic dataset with four clusters

(a) Isomap (b) modularity embedding

Fig. 2. 1-dimensional embedding of the synthetic dataset



(a) Isomap (b) modularity embedding

Fig. 3. 2-dimensional embedding of the synthetic dataset

and can't be separated. In 2-dimensional embedding, situations were improved. But still a large part of black and red clusters overlapped.

Comparatively, the modularity embedding reported significantly improved results in Fig. 2(b) and 3(b). In 1-dimensional embedding, a much smaller part of black and red data points were mixed, while most other points can be easily separated into correct clusters. In 2-dimensional embedding, all the clusters can be easily identified.

4.2 MNIST Handwritten Images

We used MNIST database [19]. The dataset collects $70,000$ handwritten images of digits from 0 to 9. Each image has 28×28 pixels. We used 250 images of digits 1 to 4 respectively, with a total of $1,000$ images. A network was built by connecting each vertex to its 6 nearest neighbors by Euclidean distances.

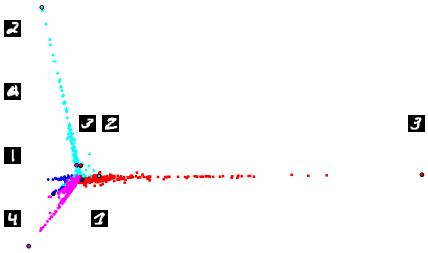


Fig. 4. 2-dimensional embedding of MNIST images

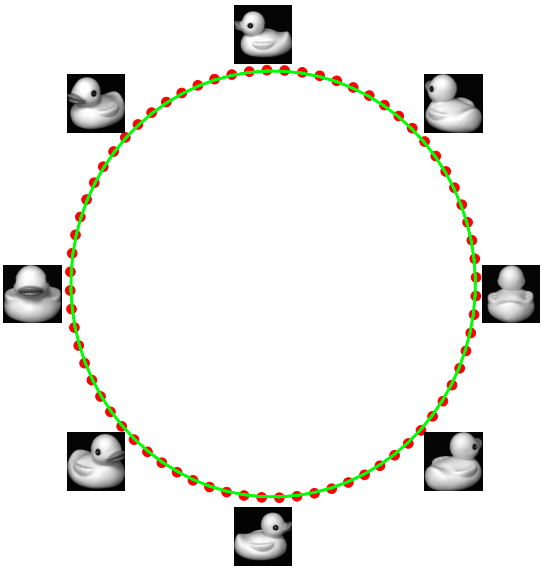


Fig. 5. 2-dimensional embedding of COIL-20 images

From the 2-dimensional embedding result in Fig. 4, we can see that the boundaries were made up with the writings that may not be easy to distinguish. Those *standard* writings were pushed far away from the boundary of the clusters.

4.3 On COIL-20 Images

In previous examples, the networks to be analyzed all have an inherent natural cluster structure. Now we'd like to investigate the performance of the proposed method if the data have no cluster structures.

We used an object's images from COIL-20 database [20]. The images were created by viewing the same object from 72 different angles, with each differing 5 degrees. Each image has 128×128 greyscale pixels, thus having an input with 16384-dimension. Although the input is very high-dimensional, the object's

images are effectively parameterized by one degree of freedom – the angle of rotation.

To apply our method, we created an undirected network with 72 vertices for all images. Each vertex was connected to its 6 nearest neighbor vertices in Euclidean distance and thus we had a connected graph. All the edge weights in the graph were set to be identically one. After applying the modularity embedding method for 2-dimensional embedding, we discovered an intuitive representation as a circle shown in Fig. 5, which effectively captures the underlying dimensionality of the images.

5 Conclusion

Significant achievements have been witnessed in the study of graph embedding methods during the recent years. These methods are based on rather different geometric intuitions, have different properties and are suitable for different applications. In this manuscript, we propose a novel modularity embedding method for graph embedding. Comparing with other methods, the new method highlights the cluster structure inherent in the data.

Acknowledgments. The work is supported by The Science and Technology Development Fund, Macao SAR, China.

References

1. Jolliffe, I.: Principal Component Analysis. Springer (2002)
2. Cox, T., Cox, M.: Multidimensional Scaling. Chapman and Hall (2000)
3. Chung, F.: Spectral Graph Theory. American Mathematical Society (1997)
4. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
5. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
6. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in Neural Information Processing Systems, vol. 14, pp. 585–591. MIT Press (2001)
7. Donoho, D.L., Grimes, C.: Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. Proceedings of the National Academy of Sciences 100(10), 5591–5596 (2003)
8. Weinberger, K.Q., Saul, L.K.: Unsupervised learning of image manifolds by semidefinite programming. International Journal of Computer Vision 70(1), 77–90 (2006)
9. Schölkopf, B., Smola, A., Müller, K.R.: Kernel principal component analysis. In: Gerstner, W., Hasler, M., Germond, A., Nicoud, J.-D. (eds.) ICANN 1997. LNCS, vol. 1327, pp. 583–588. Springer, Heidelberg (1997)
10. Nesterov, Y.E., Nemirovsky, A.S.: Interior Point Polynomial Methods in Convex Programming: Theory and Algorithms. SIAM Publishing (1993)

11. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Review 38(1), 49–95 (1996)
12. Goemans, M.X.: Semidefinite programming in combinatorial optimization. Mathematical Programming 79, 143–161 (1997)
13. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. Journal of Machine Learning Research 5, 27–72 (2004)
14. Xu, L., Neufeld, J., Larson, B., Schuurmans, D.: Maximum margin clustering. In: Advances in Neural Information Processing (2004)
15. Newman, M.: Networks: An Introduction. Oxford University Press (2010)
16. Li, W., Schuurmans, D.: Modular community detection in networks. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 1366–1371 (2011)
17. Li, W.: Revealing network communities with a nonlinear programming method. Information Sciences 229, 18–28 (2013)
18. Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z., Wagner, D.: On modularity clustering. IEEE Trans. Knowl. Data Eng. 20(2), 172–188 (2008)
19. Le Cun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of IEEE 86(11), 2278–2324 (1998)
20. Nene, S., Nayar, S., Murase, H.: Columbia object image library (COIL-20). Technical report, Columbia University (1996)

Modularity Segmentation

Wenye Li

Macao Polytechnic Institute,
Rua de Luís Gonzaga, Macao SAR, China
wyli@ipm.edu.mo
http://staff.ipm.edu.mo/~wyli

Abstract. Graph-partition based algorithms are widely used for image segmentation. We propose an improved graph-partition segmentation method based on a key notion from complex network analysis: *partition modularity*. In particular, we show how optimizing the modularity measure can automatically determine the number of segments as well as their respective structure—greatly reducing the level of human intervention in the image segmentation process. We furthermore develop an efficient spectral approach that allows for a fast segmentation procedure. The proposed method is simple, efficient, and provides a practical tool for analyzing real-world images.

Keywords: Image Segmentation, Graph Cut, Modularity Segmentation.

1 Introduction

Image segmentation involves classifying pixels into disjoint sets that correspond to individual surfaces, objects, etc. Such segmentation requires that each pixel be assigned a label such that pixels with the same label share important visual characteristics while those with differing labels do not. In this way, an image can be mapped into a simplified representation that enables easier analysis [1, 2].

Among image segmentation methods, graph-partition based approach proved to be particularly popular, primarily due to their efficiency and flexibility. In this approach an image is modeled as a graph, where vertices represent individual pixels and weighted edges represent the pairwise similarity between neighboring pixels. The graph is then partitioned into disjoint sets by optimizing standard criterion, such as minimum cut, normalized cut, or related variants [3–7]. Crucially, the number of sets in the partition is determined manually by a user in these approaches.

In this paper, we propose a new graph partition method, *modularity segmentation*, that exploits recent ideas from network analysis [8]. A key feature of the method is that it *automatically* determines the number of segments to use by optimizing a natural and theoretically justified criterion, eliminating the need for human intervention. In addition, we develop a simple and fast spectral approach to optimizing the proposed modularity measure, based on an iterative rounding approach that can be scaled to graphs with millions of vertices—a standard size for image segmentation problems.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 100–107, 2013.

2 Modularity-Based Image Segmentation

We investigate a new graph-based image segmentation method that exploits the alternative concept of *modularity* [8]. This measure arose from ground state analysis of spin glasses in theoretical physics [9], and has since been successfully applied to community detection problems in complex networks [10, 11].

2.1 The Modularity Model

In graph-partition based segmentation, each image is represented as an undirected graph $G = (V, E)$, where each vertex in $V = (v_1, v_2, \cdots, v_n)$ corresponds to an individual pixel, and each edge in E connect pairs of vertices (neighboring pixels). The weight on each edge, w_{ij}, is a nonnegative value that measures the affinity between two vertices v_i and v_j; a higher affinity indicates a stronger relation between the associated pixels. We further let $d_i = \sum_j w_{ij}$ denote the sum of the affinities associated with vertex v_i, and $m = \frac{1}{2}\sum_i d_i = \frac{1}{2}\sum_{ij} w_{ij}$ denote the total sum of the edge weights in the graph.

Given a candidate division of vertices into disjoint groups, the *modularity* is defined to be the fraction of the affinities that fall within the given groups, minus the expected such fraction when the affinities are distributed randomly. The randomization is conducted by preserving the total affinity d_i of each vertex. Under this assumption, the expected affinity between two vertices v_i and v_j is $d_i d_j / 2m$, hence the corresponding modularity is $w_{ij} - d_i d_j / 2m$. Summing over all vertex pairs within the same group, the modularity, denoted Q, is given by

$$Q = \frac{1}{2m} \sum_{ij} \left[w_{ij} - \frac{d_i d_j}{2m} \right] \delta(c_i, c_j), \tag{1}$$

where c_i denotes the group to which v_i belongs and $\delta(c_i, c_j)$ is 1 if $c_i = c_j$ and 0 otherwise. An equivalent formulation can be given by defining s_{ik} to be 1 if vertex v_i belongs to group k and 0 otherwise. Then $\delta(c_i, c_j) = \sum_k s_{ik} s_{jk}$ and hence

$$Q = \frac{1}{2m} \sum_{ij} \sum_{k} \left[w_{ij} - \frac{d_i d_j}{2m} \right] s_{ik} s_{jk} = \frac{1}{2m} tr\left(S^T B S\right), \tag{2}$$

where S is a matrix having elements s_{ik} and B is a *modularity matrix* with elements $b_{ij} = w_{ij} - d_i d_j / 2m$.

2.2 Automatic Cluster Number Discovery

The modularity measure possesses a striking set of properties that prove useful. Its value always lies in the range $[-1, 1]$. All rows and columns of the modularity matrix B sum to zero, meaning that the modularity value of an unsegmented graph is always zero. Its value is positive if the intra-group affinities exceed the expected affinities achieved at random. Thus seeking for a partition that maximizes the modularity automatically determines the appropriate number of segments to choose as well as their respective structure, rather than require the number of segments to be pre-specified.

3 Computational Approach

Unfortunately, like normalized cut, finding a graph partition that maximizes the modularity is NP-hard [12]. Therefore, approximate solutions must be sought to ensure tractability. For small to medium graphs, simulated annealing or mathematical programming methods can be applied [13–15]. For larger graphs, the dominant approach is a spectral relaxation method [16]. By computing the leading eigenvector of the modularity matrix (plus exchange heuristics), this method is able to bi-partition graphs of moderate size while attaining reasonable modularity values. However, for image segmentation, which typically involves graphs with millions of vertices, the approach still does not scale sufficiently well. Therefore, we develop a simple yet efficient new approach that enables practical applications of modularity segmentation.

3.1 A Spectral Approach

To find a segmentation that achieves high modularity in a large graph, we begin with a simple spectral method like the one proposed in [16]. First assume the graph is to be divided into just two groups, hence we can use $s_i = \pm 1$ to indicate the segment to which node v_i belongs. This leads to:

$$Q = \frac{1}{4m} \sum_{ij} b_{ij} s_i s_j = \frac{1}{4m} s^T B s, \tag{3}$$

where s is a vector with elements s_i. Now express s as a linear combination of the normalized eigenvectors, u_i, of the modularity matrix B; that is, let $s = \sum_{i=1}^{n} a_i u_i$ with $a_i = u_i^T s$. Then we have

$$Q = \frac{1}{4m} \sum_{i} a_i u_i^T B \sum_{j} a_j u_j = \frac{1}{4m} \sum_{i=1}^{n} \left(u_i^T s\right)^2 \lambda_i, \tag{4}$$

where λ_i is an eigenvalue of B with the eigenvector u_i.

Assume the eigenvalues are labeled non-increasingly, $\lambda_1 \geq \cdots \geq \lambda_n$. To maximize Q, one would like to have s concentrate as much weight as possible in the terms of the sum with the largest eigenvalues. Without "± 1" constraint, one could simply choose s to be proportional to the first eigenvector u_1, which places all the weight in the term with the largest eigenvalue λ_1. However, with the constraint, s cannot be so freely chosen, which makes the optimization difficult. In practice one often uses a simple heuristic by setting

$$s_i = \begin{cases} +1 & u_{1i} > 0 \\ -1 & otherwise \end{cases}, \tag{5}$$

which ignores the hard constraints and partitions the vertices according to the signs of each element in u_1. This is the standard spectral approach. This method can be easily applied by using a power iteration method [17] to efficiently compute the dominant eigenvalue and eigenvector of B.

The spectral method directly works for two-way partitions. For multi-way partitions, a recursive method is possible by applying and checking the two-way partition results repeatedly on subgraphs.

In practice, the spectral method is applied in conjunction with an *exchange heuristic* [18]. Given two groups of vertices, a refinement successively finds the vertex that, when moved to the other group, achieves the largest increase in Q, or the smallest decrease if no increase exists. Such moves are repeated until no more improvement is possible. Unfortunately, the heuristic has an expensive cost, nearly $O\left(n^3\right)$, which prevents the spectral method's application to large problems as in image segmentation. We have to seek other solutions.

3.2 Iterative Rounding

To overcome the difficulties in maximizing the partition modularity, we propose an iterative rounding approach that avoids the complexity of exchange heuristics with ideas rooted in [19, 20].

Recall that the standard spectral method uses a rounding strategy, Eq. (5), entirely based on the signs of the leading eigenvector elements, regardless of their magnitudes. However, it is clear that different magnitude contributes differently to Q, and hence affects our confidence in making decisions. If u_{1i} has a large magnitude, then s_i has significant influence on $u_1^T s$ and therefore on Q, one would be more confident to infer its rounded value. Conversely, if the magnitude is small, s_i's contribution is not evident, and one would be less confident. In this case, we postpone the rounding to a later phase.

Based on this simple intuition, we propose a successive rounding approach that only rounds variables with large magnitudes. That is, unlike conventional rounding that sets the entire bi-partition in a single batch, we propose to recover a more accurate partition structure incrementally, using a strategy we refer to as *iterative rounding*.

Consider the following illustration of the proposed approach. In the first iteration, one has the same problem, $\max s^T B s$, as the standard spectral method. Here, one simply uses the power method to obtain the initial leading eigenvector. However, rather than deploy conventional rounding, we then only round the elements with sufficiently large magnitudes (say, ≥ 1) into decisions.

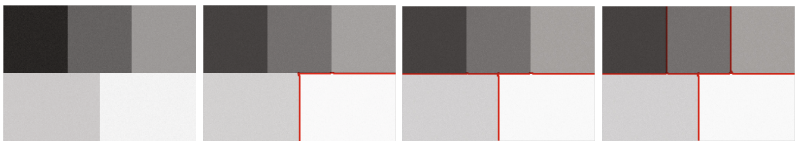
After the first iteration, we are left a residual problem to solve. Now suppose $s = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$ where s_1 denotes the elements that are already fixed, and s_2 has γ elements yet to be rounded to ± 1. The objective is then to maximize

$$\begin{pmatrix} s_1 \\ s_2 \end{pmatrix}^T \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = s_2^T B_{22} s_2 + 2 s_2^T B_{21} s_1 + s_1^T B_{11} s_1 \tag{6}$$

where B_{11}, B_{12}, B_{21} and B_{22} are sub-matrices of B. Equivalently, we maximize

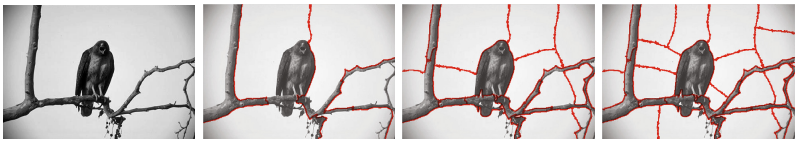
$$L = s_2^T B_{22} s_2 + 2 s_2^T B_{21} s_1 \tag{7}$$

with respect to s_2, subject to the constraint that $\|s_2\| = \sqrt{\gamma}$.



(a) (b) $Q = .3774$ (c) $Q = .6271$ (d) $Q = .7906$

Fig. 1. (a): Synthetic gray-scale image. (b)-(d): Different number (2,3,5) of segments separated by bold lines, showing the corresponding modularity values achieved.



(a) (b) $Q = .4958$ (d) $Q = .8451$ (e) $Q = .8885$

Fig. 2. (a): Natural gray-scale image. (b)-(d): Different number (2, 8, 18) of segments separated by bold lines, showing the corresponding modularity values achieved.

Although the residual problem appears harder than the initial problem, an efficient solution procedure fortunately exists. The main idea is to employ a generalized form of power method iteration: Starting with s_2^0 (which may be $B_{21}s_1$ provided it is not zero, or a random vector), successively apply the update

$$s_2^{t+1} = \frac{B_{22}s_2^t + B_{21}s_1}{\|B_{22}s_2^t + B_{21}s_1\|}\sqrt{\gamma} \tag{8}$$

until convergence.

This update can be explained intuitively. For s_2 to reach a fixed point the gradient ∇L must be parallel to s_2: $\nabla L = 2\lambda s_2$ for some scalar λ, yielding

$$s_2 = \frac{B_{22}s_2 + B_{21}s_1}{\lambda}. \tag{9}$$

Considering the length requirement on s_2, it must hold that $\|B_{22}s_2 + B_{21}s_1\| = \lambda\sqrt{\gamma}$, hence λ must be positive and we recover the update rule above.

4 Evaluation

To evaluate the proposed model, we applied it to both synthetic and real images from the Berkeley segmentation dataset.Each image has 481×321 gray-scale pixels, which yields a graph of $154,401$ vertices. The graph affinity matrix W was constructed by the intervening contours method [21].

4.1 Detecting the Number of Segments

A synthetic example helps illustrate the capability of modularity segmentation in detecting the correct number of segments. Fig. 1(a) depicts a synthetic image

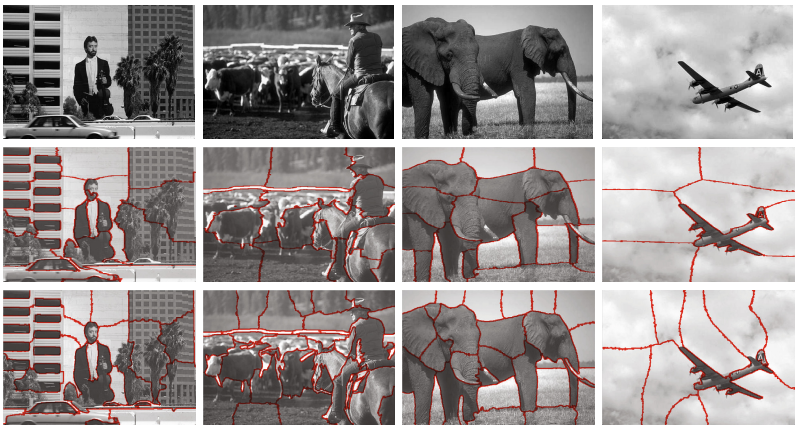


Fig. 3. The first row shows the original images. The second row shows the segmentation results by normalized cut. The last row shows the modularity segmentation results.

with four regions of different intensities to which zero-mean white noise with a standard deviation of 0.02 was added. After a two-way segmentation, the modularity value achieved is 0.3774 (Fig. 1(b)). With more segments, the modularity value increases to 0.6271 (Fig. 1(c)) and stops at its maximum of 0.7906 with five segments (Fig. 1(d)).

Fig. 2 demonstrates the proposed method's behavior on a natural scene. With a binary partition (Fig. 2(b)), the modularity value is 0.4958. The value increases with additional segments, finally reaching a maximum of 0.8885 at eighteen segments (Fig. 2(d)). For these two examples, modularity segmentation correctly identifies major components of the scene.

4.2 Comparison to Normalized Cut

To compare the performance of modularity segmentation to the state-of-the-art, we conducted a number of experiments with normalized cut. In particular, we ran modularity segmentation to first recover the number of segments that achieves the highest Q value, then ran normalized cut with the same number of segments. Fig. 3 demonstrates typical outcomes from this experiment. Here we can see that the two methods achieve similar segmentation quality on the benchmark images. These results held throughout our evaluation on the Berkeley dataset.

4.3 Time and Memory Requirements

We also studied the time and memory requirements of the various methods. All methods were implemented in MATLAB [1], and ran on a workstation with a single Intel i7-980 CPU and 24GB RAM.

[1] An efficient MATLAB implementation of normalized cut was downloaded from `http://www.seas.upenn.edu/~timothee`.

In comparing the time required by normalized cut and modularity segmentation, we observed that for small images of 50×50 pixels, the two methods had comparable speed and partition images in less than a second. As the image size increased, the two methods demonstrated similar scaling in run time; on images of 1000×1000 pixels, both required about 15 minutes.

For the memory consumed, normalized cut must compute the eigenvectors of an affinity matrix, which is sparse and memory therefore is not a significant concern. In comparison, modularity segmentation needs to compute the first eigenvector of a dense modularity matrix. Fortunately, by utilizing its relationship with the affinity matrix (cf. Section 2.1), one does not need to store the dense matrix either. In particular, modularity segmentation shows similar trends in memory consumption to normalized cut, as our experiments verified.

5 Conclusion

Our work investigates the possibility of applying a criterion of network modularity to image segmentation. Unlike previous graph partition methods, maximizing the modularity measure automatically determines the appropriate number of segments to use, which reduces the level of manual intervention required to perform image segmentation. We have proposed a specialized spectral method and an accompanying iterative rounding strategy that yields efficient yet accurate results on large scale problems, enabling the use of modularity in image processing. The outcome is an effective segmentation technique that achieves state-of-the-art quality and efficiency while automating an important facet of the segmentation process.

Acknowledgments. The work is supported by The Science and Technology Development Fund, Macao SAR, China. The author acknowledges the beneficial discussions with Prof. Dale Schuurmans in University of Alberta.

References

1. Forsyth, D., Ponce, J.: Computer Vision: A Modern Approach. Prentice-Hall (2003)
2. Shapiro, L.G., Stockman, G.C.: Computer Vision. Prentice-Hall (2001)
3. Grady, L.: Random walks for image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 28(11), 1768–1783 (2006)
4. Shi, J.B., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
5. Witkin, A.P.: Scale-space filtering. In: Proceedings of the 8th International Joint Conference on Artificial Intelligence, pp. 1019–1022 (1983)
6. Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 15(11), 1101–1113 (1993)
7. Zahn, C.T.: Graph-theoretical methods for detecting and describing gestalt clusters. IEEE Trans. Computer 20(1), 68–86 (1971)

8. Newman, M., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69, 26113 (2004)
9. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. Physical Review E 74 (2006)
10. Easley, D., Kleinberg, J.: Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press (2010)
11. Newman, M.: Networks: An Introduction. Oxford University Press (2010)
12. Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z., Wagner, D.: On modularity clustering. IEEE Trans. Knowl. Data Eng. 20(2), 172–188 (2008)
13. Agarwal, G., Kempe, D.: Modularity-maxmizing graph communities via mathematical programming. European Physical Journal B 66 (2008)
14. Guimerá, R., Amaral, L.: Cartography of complex networks: modules and universal roles. Journal of Statistical Mechanics (2005)
15. Li, W.: Revealing network communities with a nonlinear programming method. Information Sciences 229, 18–28 (2013)
16. Newman, M.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences 103(23), 8577–8582 (2006)
17. Reiter, C.: Easy algorithms for finding eigenvalues. Mathematics Magazine 63(3), 173–178 (1990)
18. Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the traveling salesman problem. Operations Research 21, 498–516 (1973)
19. Li, W., Schuurmans, D.: Modular community detection in networks. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 1366–1371 (2011)
20. Xu, L., Li, W., Schuurmans, D.: Fast normalized cut with linear constraints. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2866–2873 (2009)
21. Malik, J., Belongie, S.J., Leung, T., Shi, J.B.: Contour and texture analysis for image segmentation. International Journal of Computer Vision 43(1), 7–27 (2001)

Brand-Centric Recommendation with Inter-brand Similarities

Hyojung Shin[1] and Seungjin Choi[2]

[1] Microsoft SQL Server, Enterprise & Tier 1
3940 159th Avenue Northeast, Redmond, WA, USA
hotcoa@gmail.com
[2] Department of Computer Science and Engineering
Pohang University of Science and Technology
77 Cheongam-ro, Nam-gu, Pohang 790-784, Korea
seungjin@postech.ac.kr

Abstract. With the increasing popularity of mobile web, social networking services (SNSs) are an integral part of our everyday lives, since they are used for communicating with friends, for gaining information on other people or some items of interest, and even for business profits. Social network information is incorporated into recommender systems to improve their performance, but most of existing work is focused on user-centric cases in which items or venues are recommended for users. On the other hand, brands are also important social objects. For instance, in Foursquare, which is a location-based online SNS, brands provide information on venues and share the tips with their followers. Thus, it is important to recommend venues for brands so that brands select interesting venues for their followers, leading to *brand-centric recommendation* where the targets for recommendation are brands (not users). In contrast to user-centric recommendation, brands have few social links to other brands, so trust between brands is difficult to use. In this paper we present a method for brand-centric recommendation where *inter-brand similarities* are implicitly determined by decomposing a brand-follower matrix. This social information on inter-brand similarity is incorporated into probabilistic matrix factorization to reveal brand latent factors as well as venue latent factors. Experiments on the dataset collected from Foursquare (by web crawling) demonstrate that our method improves the recommendation performance over existing matrix factorizations.

Keywords: Collaborative filtering, matrix factorization, recommender systems.

1 Introduction

As social network services have been widely used, there emerged diverse types of users. Brands, communities and celebrities started to use social networking services and their proportions are becoming large. Brands can be corporations, chain-stores, online services, famous TV-series or others. Users follow interesting

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 108–115, 2013.

brands and get related information. For example, in Foursquare[1], a location-based online social network application, brands give tips on venues and followers check the places and tips. Online dating sites present good restaurants or shopping sites shows great places to buy clothes. Also, in Facebook[2], brands such as famous TV-series or movies update famous quotes everyday or advertise related events. It is time that brands need to extend their stage to online world to sell their products and increase brand value. For brands to effectively get popular, there needs recommender system which predicts their followers' interests, brand-centric recommendation.

Matrix factorization is a method for seeking a low-rank latent structure of data, approximating the data matrix as a product of two or more factor matrices. Matrix factorization is popular for collaborative prediction, where unknown ratings are predicted by user and item factor matrices which are determined to approximate a user-item matrix as their product [1–3]. Various methods for trust-aware recommendation have been developed to predict ratings of users while taking rating values of friends as an independent source of information [4, 5]. However, existing methods cannot be directly applied to brand-centric recommendation because brands have a few social links with other brands. The goal of brands is to attract many people and advertise to them, not to socialize with other brands. Thus, we infer inter-brand similarities implicitly by decomposing brand-follower matrix constructed from a social network of brands and their followers. Then, we incorporate these inter-brand similarities into the trust-exploited matrix factorization method [5]. Experiments on the Foursquare dataset (by web crawling) demonstrate that our method improves the recommendation performance over existing matrix factorizations.

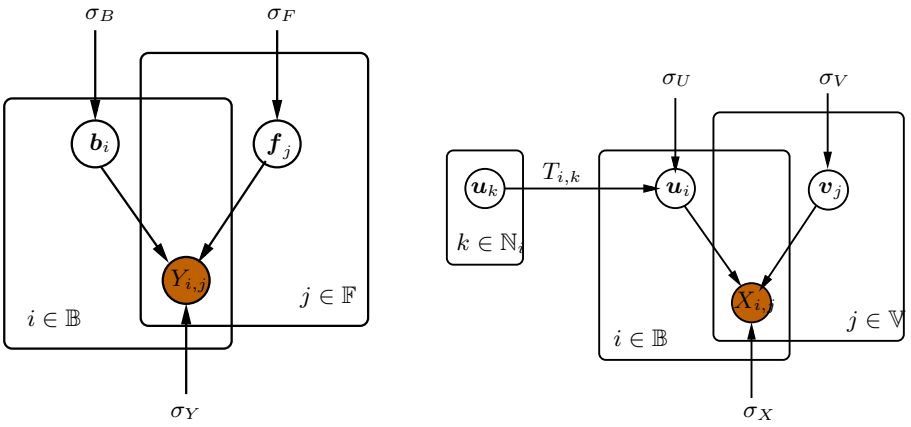
2 Method

Denote by $\mathbb{B}$, $\mathbb{V}$, $\mathbb{F}$, a set of N brands, a set of M venues, and a set of P followers of brands, respectively. The brand-follower matrix $\boldsymbol{Y}$ is a binary matrix, in which entries $Y_{i,j} = 1$ implies brand i is followed by user j and otherwise $Y_{i,j} = 0$. Entries $X_{i,j}$ of the brand-venue rating matrix represents the rating value on venue j by brand i. Our method consists of two steps:

1. We compute inter-brand similarities $\{T_{i,k}\}$, inferring latent matrices $\boldsymbol{B}$ and $\boldsymbol{F}$ which approximates the the brand-follower matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times P}$ (see Fig. 1 (a)).
2. We then learn a weighted low-rank approximation of the brand-venue matrix $\boldsymbol{X}$, decomposing it as $\boldsymbol{X} = \boldsymbol{U}^\top \boldsymbol{V}$ with $\boldsymbol{U}$ regularized by inter-brand similarities, as shown in Fig. 1 (b).

Prediction of the rating value on a venue by a brand is to fill in missing entries of the brand-venue matrix $\boldsymbol{X}$. This is done by the reconstruction $\boldsymbol{U}^\top \boldsymbol{V}$.

[1] www.foursquare.com
[2] www.facebook.com



(a) (b)

Fig. 1. Graphical representations for matrix factorizations for: (a) brand-follower matrix; (b) brand-venue matrix with inter-brand similarities.

2.1 Inter-brand Similarity

We calculate inter-brand similarities $\{T_{i,k}\}$, approximating the brand-follower matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times P}$ as $\boldsymbol{Y} = \boldsymbol{B}^\top \boldsymbol{F}$, where $\boldsymbol{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N] \in \mathbb{R}^{K \times N}$ is the brand-specific latent matrix and $\boldsymbol{F} = [\boldsymbol{f}_1, \ldots, \boldsymbol{f}_P] \in \mathbb{R}^{K \times P}$ is the follower-specific latent matrix. Graphical representation for this factorization model is shown in Fig. 1 (a), where the probability of observing the brand-follower matrix $\boldsymbol{Y}$ conditioned on latent matrices $\boldsymbol{B}$ and $\boldsymbol{F}$ is given by

$$p(\boldsymbol{Y}|\boldsymbol{B}, \boldsymbol{F}, \sigma_Y^2) = \prod_{i=1}^{N} \prod_{j=1}^{P} \left[\mathcal{N}(Y_{i,j}|\boldsymbol{b}_i^\top \boldsymbol{f}_j, \sigma_Y^2)\right]^{O_{i,j}^Y}, \tag{1}$$

where $\mathcal{N}(y|\mu_y, \sigma_y^2)$ denotes Gaussian distribution with mean μ_y and variance σ_y^2, and $\{O_{i,j}^Y\}$ are indicators to specify observed entries of $\boldsymbol{Y}$, i.e., $O_{i,j}^Y = 1$ when $Y_{i,j}$ is observed and otherwise $O_{i,j}^Y = 0$. We place zero-mean spherical Gaussian priors on $\boldsymbol{B}$ and $\boldsymbol{F}$:

$$p(\boldsymbol{B}|\sigma_B^2) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{b}_i|0, \sigma_B^2 \boldsymbol{I}), \tag{2}$$

$$p(\boldsymbol{F}|\sigma_F^2) = \prod_{j=1}^{P} \mathcal{N}(\boldsymbol{f}_j|0, \sigma_F^2 \boldsymbol{I}), \tag{3}$$

where $\boldsymbol{I}$ is the identity matrix. As in probabilistic matrix factorization (PMF) [6], we compute MAP estimates of $\boldsymbol{B}$ and $\boldsymbol{F}$. To this end, we consider the log of

the posterior distribution over $\boldsymbol{B}$ and $\boldsymbol{F}$:

$$\log p(\boldsymbol{B}, \boldsymbol{F}|\boldsymbol{Y}) \propto -\frac{1}{2\sigma_Y^2}\sum_{i=1}^{N}\sum_{j=1}^{P} O_{i,j}^{Y}(Y_{i,j} - \boldsymbol{b}_i^\top \boldsymbol{f}_j)^2 - \frac{1}{2\sigma_B^2}\sum_{i=1}^{N}\|\boldsymbol{b}_i\|^2 - \frac{1}{2\sigma_F^2}\sum_{j=1}^{P}\|\boldsymbol{f}_j\|^2, \tag{4}$$

where $\|\cdot\|$ represents Euclidean norm. Then we calculate binary inter-brand similarities $T_{i,k}$ in accordance with cosine-similarities between MAP estimates of brand-specific latent vectors $\boldsymbol{b}_i$ and $\boldsymbol{b}_k$, given a threshold value t, i.e.,

$$T_{i,k} = \begin{cases} 1 \text{ if } \cos(\boldsymbol{b}_i, \boldsymbol{b}_k) \geq t, \\ 0 \text{ otherwise.} \end{cases} \tag{5}$$

2.2 Matrix Factorization Regularized by Inter-brand Similarity

We decompose brand-venue matrix $\boldsymbol{X}$ decomposed as $\boldsymbol{X} = \boldsymbol{U}^\top \boldsymbol{V}$, as shown in Fig. 1 (b), where inter-brand similarities are incorporated using the relation

$$\boldsymbol{u}_i = \frac{\sum_{k\in\Omega_i} T_{i,k}\boldsymbol{u}_k}{\sum_{k\in\Omega_i} T_{i,k}},$$

where Ω_i is the set of brands which are related to brand i. Our regularized matrix factorization was motivated by [5, 7], where social network information among users is incorporated into decomposing a user-rating matrix. We normalize each row of the inter-brand similarity $\boldsymbol{T}$ such that $\sum_{j=1}^{N} T_{i,j} = 1$. Then we have

$$\boldsymbol{u}_i = \sum_{k\in\Omega_i} T_{i,k}\boldsymbol{u}_k. \tag{6}$$

The joint distribution over $\boldsymbol{X}$, $\boldsymbol{U}$, and $\boldsymbol{V}$ is given by

$$p(\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{V}) = p(\boldsymbol{X}|\boldsymbol{U}, \boldsymbol{V})p(\boldsymbol{U}|\boldsymbol{T})p(\boldsymbol{U})p(\boldsymbol{V}), \tag{7}$$

where the probability of observing $X_{i,j}$ conditioned on $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ is described by

$$p(\boldsymbol{X}|\boldsymbol{U}, \boldsymbol{V}, \sigma_X^2) = \prod_{i=1}^{N}\prod_{j=1}^{M}\left[\mathcal{N}(X_{i,j}|\boldsymbol{u}_i^\top \boldsymbol{v}_j, \sigma_X^2)\right]^{O_{i,j}^X}, \tag{8}$$

where $O_{i,j}^X = 1$ when $X_{i,j}$ is observed and otherwise $O_{i,j}^X = 0$. We place zero-mean Gaussian priors on both $\boldsymbol{U}$ and $\boldsymbol{V}$:

$$p(\boldsymbol{U}|\sigma_U^2) = \prod_{i=1}^{N}\mathcal{N}(\boldsymbol{u}_i|0, \sigma_U^2\boldsymbol{I}), \tag{9}$$

$$p(\boldsymbol{V}|\sigma_V^2) = \prod_{j=1}^{M}\mathcal{N}(\boldsymbol{v}_j|0, \sigma_V^2\boldsymbol{I}). \tag{10}$$

The conditional distribution over brand-specific latent matrix $\boldsymbol{U}$ given its direct neighbors is

$$p(\boldsymbol{U}|\boldsymbol{T}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{u}_i | \sum_{k \in \Omega_i} T_{i,k}\boldsymbol{u}_k, \sigma_T^2 \boldsymbol{I}). \tag{11}$$

With this parametrization, the log of the posterior distribution over $\boldsymbol{U}$ and $\boldsymbol{V}$ is given by

$$\begin{aligned} \log p(\boldsymbol{U}, \boldsymbol{V}|\boldsymbol{X}) \propto & -\frac{1}{2\sigma_X^2} \sum_{i=1}^{N} \sum_{j=1}^{M} O_{i,j}^X (X_{i,j} - \boldsymbol{u}_i^\top \boldsymbol{v}_j)^2 - \frac{1}{2\sigma_T^2} \sum_{i=1}^{N} \|\boldsymbol{u}_i - \sum_{k \in \Omega_i} T_{i,k}\boldsymbol{u}_k\|^2 \\ & - \frac{1}{2\sigma_U^2} \sum_{i=1}^{N} \|\boldsymbol{u}_i\|^2 - \frac{1}{2\sigma_V^2} \sum_{j=1}^{M} \|\boldsymbol{v}_j\|^2, \end{aligned} \tag{12}$$

leading to the following objective function

$$\begin{aligned} \mathcal{J} = & \sum_{i=1}^{N} \sum_{j=1}^{M} O_{i,j}^X (X_{i,j} - \boldsymbol{u}_i^\top \boldsymbol{v}_j)^2 + \lambda_T \sum_{i=1}^{N} \|\boldsymbol{u}_i - \sum_{k \in \Omega_i} T_{i,k}\boldsymbol{u}_k\|^2 \\ & + \lambda_U \sum_{i=1}^{N} \|\boldsymbol{u}_i\|^2 + \lambda_V \sum_{j=1}^{M} \|\boldsymbol{v}_j\|^2, \end{aligned} \tag{13}$$

where $\lambda_T = \sigma_X^2/\sigma_T^2, \lambda_U = \sigma_X^2/\sigma_U^2, \lambda_V = \sigma_X^2/\sigma_V^2$. We use stochastic coordinate descent to determine $\boldsymbol{U}$ and $\boldsymbol{V}$ which mimimize (13).

3 Experiments

3.1 Foursquare Dataset Description

We constructed 'Foursquare Dataset' crawled from Foursquare which is a location-based social networking website launched in 2009. Users access Foursquare through mobile devices and check-in at places. Users also write tips about venues which serve as suggestions for great things to do, see or eat at the venue. They create to-do lists and add interesting venues to visit or tips to follow next time. Users can follow others or have reciprocal relationships with others. There are three types of users: *brand*, *celebrity* and *user*. *Brand* is a type of users for corporations, chain-stores, online-services and other communities. *Celebrity* is for TV stars, singers or people with more than 1000 friends. *User* is a people who uses the service for information on interesting venues and communication with their friends. For our dataset, we crawled all brands in Foursquare and their followers. We also crawled the venues where the brands have given tips on. Each tip provides the number of people who added the tips to their to-do list, *todo* count, and the number of people who followed the tips, *done* count.

Table 1 shows the statistics of the dataset in the left panel, in which the number of brands, venues, followers, tips by brands, and links that relate users to brands, is shown. To avoid the sparsity problem, we considered brands which have given tips on at least two venues and venues which have at least two tips. The right panel in Table 1 shows the distributions of the dataset. A brand gave tips on 8 venues on average and 147 venues at maximum. Minimum number of followers of a brand is 9, average number is 315 and maximum number is 499.

Table 1. Description of Foursquare dataset

number of brands	1832
number of venues	5437
number of followers	35113
number of tips by brands	14739
number of links	577681

Avg. venues of tips	8
Max. venues of tips	147
Min. followers	9
Avg. followers	315
Max. followers	499

The total counts of *todo* and *done* can be interpreted as a rating value of a brand on a venue since high number of counts would imply the popularity of the venue among followers. For rating matrix X, we transformed the summation of *todo* and *done* counts into ratings on a scale from 1 to 5. Since some brands might get high number of counts because it is already popular, we need to scale differently according to brands. We use the method of *equal width intervals*(EWI) [8]. Let $count(i, j)$ be the summation of *todo* and *done* counts of a tip where a brand i gave on venue j. Let $count(i)$ be the set of the number of counts of tips given by brand i. First, EWI computes a step for each brand i as:

$$step_i = \frac{\max(count(i)) - \min(count(i))}{5}. \tag{14}$$

The rating for brand i on venue j, $X_{i,j}$, is set to $k \in \{1, 2, 3, 4, 5\}$ if,

$$step_i * (k-1) <= count_{i,j} - Min(count(i)) < step_i * k. \tag{15}$$

3.2 Empirical Results

For comparisons, we compare TrustMF with the following methods.

- *UserMean*: For $X_{i,j}$, this recommends average ratings of brand i.
- *ItemMean*: For $X_{i,j}$, this recommends average ratings of venue j.
- *PMF*: This is probabilistic matrix factorization which trains latent features based solely on rating matrix X.

The parameters are set as $\lambda_T = \lambda_U = \lambda_V = 0.1$ and $K = 5$. The threshold value for constructing trust network is set to 0.9.

For the performance measurements, we used RMSE and MAE.

$$RMSE = \sqrt{\frac{\sum_{(i,j)\in X_{test}} (X_{i,j} - \hat{X}_{i,j})^2}{|X_{test}|}}, \tag{16}$$

$$MAE = \frac{\sum_{(i,j)\in X_{test}} |X_{i,j} - \hat{X}_{i,j}|}{|X_{test}|}, \tag{17}$$

where $X_{i,j}$ is true rating value, $\hat{X}_{i,j}$ is predicted value and X_{test} is dataset for test.

For experiments, we prepared two datasets. The Dataset1 has test dataset of ratings given by cold-start brands and Dataset2 has test dataset from ratings given by brands who have less than 20 friends from trust network T. Cold-start brands are those with less than 5 ratings.

Table 2. The RMSE and MAE values on Dataset1

Method	RMSE	MAE
UserMean	2.0604(15.06%)	1.6470(25.03%)
ItemMean	2.0593(15.01%)	1.6491(25.12%)
PMF	1.8197(3.82%)	1.3149(6.10%)
TrustMF	1.7501(-)	1.2347(-)

Table 2 shows the RMSE and MAE results on Dataset1. It reports that TrustPMF enhances RMSE and MAE significantly compared to other methods. TrustMF enhances RMSE of UserMean, ItemMean and PMF by 15.06%, 15.01% and 3.82% each. Also, it enhances MAE of UserMean, ItemMean and PMF by 25.03%, 25.12% and 6.10% each. The results verify that cold-start brands get benefit from the trust network. One noticeable point is that MAE, compared to RMSE, shows more significant improvement. RMSE gets largely affected by big differences of ratings and tends to ignore small improvements, whereas MAE takes values of errors with equal weights. Though TrustMF does not significantly reduce the errors with big gap, it predicts ratings precisely in overall.

Table 3. The RMSE and MAE values on Dataset2

Method	RMSE	MAE
UserMean	2.0176(37.50%)	1.7074(59.04%)
ItemMean	2.0154(37.44%)	1.7049(58.98%)
PMF	1.2740(1.02%)	0.7450(6.12%)
TrustPMF	1.2609(-)	0.6994(-)

The RMSE and MAE values on Dataset2 are presented in Table 3. Apparently, TrustMF shows significantly accurate results compared to UserMean and ItemMean. Compared to PMF, TrustMF enhances RMSE and MAE by 1.02%

and 6.12%. As previously seen in Dataset1, MAE shows more improvement than RMSE. The results prove that the inferred trust network T helps improve recommendation accuracies.

4 Conclusions

We have addressed a problem of brand-centric recommendation which is the first attempt in recommendation systems, to our best knowledge. To this end, we have developed a method of computing inter-brand similarities where the brand-follower matrix is decomposed to infer brand-specific latent factors which are used to calculate inter-brand similarities. These inter-brand similarities are incorporated into the matrix factorization involving the brand-venue matrix. Experiments on 'Foursquare Dataset' confirmed the validity of our method.

Acknowledgments. This work was supported by NIPA ITRC Support Program (NIPA-2013-H0301-13-3002), and POSTECH Rising Star Program.

References

1. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Computer 4242(8), 30–37 (2009)
2. Yoo, J., Choi, S.: Bayesian matrix co-factorization: Variational algorithm and Cramér-Rao bound. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD), Athens, Greece (2011)
3. Park, S., Kim, Y.D., Choi, S.: Hierarchical Bayesian matrix factorization with side information. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Beijing, China (2013)
4. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), Boston, MA (2009)
5. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the ACM International Conference on Recommender Systems (RecSys), Barcelona, Spain (2010)
6. Salakhutdinov, R., Mnih, A.: Probablistic matrix factorization. In: Advances in Neural Information Processing Systems (NIPS), vol. 20. MIT Press (2008)
7. Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I.: Recommender systems with social regularization. In: Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM), Hong Kong (2010)
8. Berjani, B., Strufe, T.: A recommendation system for spots in location-based online social networks. In: Proceedings of the 4th Workshop on Social Network Systems, Salzburg, Austria (2011)

Flexible Nonparametric Kernel Learning with Different Loss Functions

En-Liang Hu[1,2] and James T. Kwok[1]

[1] Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
[2] Department of Mathematics
Yunnan Normal University, Yunnan, China
{jamesk,ynelhu}@cse.ust.hk

Abstract. Side information is highly useful in the learning of a nonparametric kernel matrix. However, this often leads to an expensive semidefinite program (SDP). In recent years, a number of dedicated solvers have been proposed. Though much better than off-the-shelf SDP solvers, they still cannot scale to large data sets. In this paper, we propose a novel solver based on the alternating direction method of multipliers (ADMM). The key idea is to use a low-rank decomposition of the kernel matrix $\mathbf{Z} = \mathbf{X}^\top \mathbf{Y}$, with the constraint that $\mathbf{X} = \mathbf{Y}$. The resultant optimization problem, though non-convex, has favorable convergence properties and can be efficiently solved without requiring eigen-decomposition in each iteration. Experimental results on a number of real-world data sets demonstrate that the proposed method is as accurate as directly solving the SDP, but can be one to two orders of magnitude faster.

1 Introduction

Kernel methods have been highly successful in classification, regression, clustering, ranking, and dimensionality reduction. Because of the central role of the kernel, it is important to identify an appropriate kernel function or matrix for the task at hand. Over the past decade, there have been a large body of literature on this kernel learning problem [8,1]. While a parametric form of the kernel or a combination of multiple kernels are often assumed, nonparametric kernel learning, which takes no such assumptions, is more flexible and has received significant interest in recent years [10,7,6,14,11].

To facilitate kernel learning, obviously one has to utilize information from the data. The most straightforward approach is to use class labels. However, obtaining label information may sometimes be expensive and time-consuming. In this paper, we focus on a weaker form of supervisory information, namely, the so-called *must-link* and *cannot-link* pairwise constraints [12]. These pairwise constraints, or *side information*, define whether the two patterns involved should belong to the same class or not. Another useful source of information, which is commonly used in semi-supervised learning, is the data manifold [2]. This encourages patterns that are locally nearby on the manifold to have similar predicted labels.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 116–123, 2013.

To learn a kernel matrix $\mathbf{Z}$, we consider the following SDP problem

$$\min_{\mathbf{Z}\succeq 0} \mathscr{R}(\mathbf{Z}) + \lambda \ell(\mathbf{Z}, \mathbf{T}), \tag{1}$$

where $\mathbf{Z} = [Z_{ij}] \in \mathbb{R}^{n\times n}$ is the kernel matrix to be learned (which has to be symmetric and positive semidefinite (psd), denoted $\mathbf{Z} \succeq \mathbf{0}$), $\mathbf{L}$ is the graph Laplacian matrix of the data manifold, $\mathbf{T} = [T_{ij}]$ with a similarity/dissimilarity (resp. *must-link*/*cannot-link*) indicator matrix such that $T_{ij} = \begin{cases} 1 & (i,j) \in \mathcal{S}, \\ -1 & (i,j) \in \mathcal{D}, \end{cases}$, and λ is a regularization parameter. As in [6,9,11], we use a low-rank approximation on the kernel matrix $\mathbf{Z}$. In other words, $\mathbf{Z}$ is approximated as $\mathbf{Z} \simeq \mathbf{X}^\top \mathbf{X}$, where $\mathbf{X} \in \mathbb{R}^{r\times n}$ and rank $r \ll n$. Problem (1) can then be rewritten as

$$\min_{\mathbf{X},\mathbf{Z}} \mathscr{R}(\mathbf{Z}) + \lambda \ell(\mathbf{Z}, \mathbf{T}) \ : \ \mathbf{Z} = \mathbf{X}^\top \mathbf{X}. \tag{2}$$

In this paper, we present a novel solver for (2) based on the alternating direction method of multipliers (ADMM) [3]. Our key observation is that (2) can often be decoupled as $\sum_{ij} \left(\tilde{\mathscr{R}}(Z_{ij}) + \tilde{\ell}(Z_{ij}, T_{ij}) \right)$. Thus, solving (2) reduces to the solving of each individual entry Z_{ij}, which is easier and more efficient.

Notations: In the sequel, matrices and vectors are denoted in bold, with upper-case letters for matrices and lower-case for vectors. The transpose of a vector/matrix is denoted by the superscript $^\top$. Moreover, $\mathbf{I}$ is the identity matrix.

2 Alternating Direction Method of Multipliers (ADMM)

ADMM is a simple but powerful algorithm that has been successfully used in machine learning and data mining. The standard ADMM is for solving convex problems. Here, we consider the more general bi-convex problem [3]:

$$\min_{\mathbf{x},\mathbf{y}} F(\mathbf{x}, \mathbf{y}) \ : \ G(\mathbf{x}, \mathbf{y}) = \mathbf{0}, \tag{3}$$

where $F(\cdot,\cdot)$ is bi-convex and $G(\cdot,\cdot)$ is bi-affine[1]. As in the method of multipliers, the more general ADMM considers the augmented Lagrangian of (3): $\mathscr{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\Lambda}) = F(\mathbf{x}, \mathbf{y}) + \boldsymbol{\Lambda}^\top G(\mathbf{x}, \mathbf{y}) + \frac{\rho}{2} \|G(\mathbf{x}, \mathbf{y})\|^2$, where $\boldsymbol{\Lambda}$ is the vector of Lagrangian multipliers, and $\rho > 0$ is a penalty parameter. At the kth iteration, the values of $\mathbf{x}, \mathbf{y}$ and $\boldsymbol{\Lambda}$ (denoted $\mathbf{x}^k, \mathbf{y}^k$ and $\boldsymbol{\Lambda}^k$) are updated as

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \mathscr{L}(\mathbf{x}, \mathbf{y}^k, \boldsymbol{\Lambda}^k), \ \mathbf{y}^{k+1} = \arg\min_{\mathbf{y}} \mathscr{L}(\mathbf{x}^{k+1}, \mathbf{y}, \boldsymbol{\Lambda}^k),$$

$$\boldsymbol{\Lambda}^{k+1} = \boldsymbol{\Lambda}^k + \rho G(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}).$$

Note that while the method of multipliers minimizes $\mathscr{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\Lambda}^k)$ w.r.t. $\mathbf{x}$ and $\mathbf{y}$ jointly, ADMM allows easier decomposition of the optimization problem by minimizing them in an alternating manner.

[1] In other words, for any fixed $\mathbf{x}, \mathbf{y}$, $F(\cdot, \mathbf{y})$ and $F(\mathbf{x}, \cdot)$ are convex; while $G(\cdot, \mathbf{y})$ and $G(\mathbf{x}, \cdot)$ are affine.

3 Kernel Learning by ADMM

In this section, we introduce an extra variable $\mathbf{Y}$ to allow easier decoupling of the optimization problem. Specifically, (2) can be equivalently formulated as

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{Z}} \mathscr{R}(\mathbf{Z}) + \lambda\ell(\mathbf{Z},\mathbf{T}) \ : \ \mathbf{Z} = \mathbf{X}^\top\mathbf{Y},\ \mathbf{X} = \mathbf{Y}. \tag{4}$$

Consider $\mathscr{R}(\mathbf{Z}) = tr(\mathbf{ZL})$. Notice that both $tr(\mathbf{ZL})$ and $\ell(\mathbf{Z},\mathbf{T})$ are bi-convex, and that the constraints are bi-affine. The augmented Lagrangian of (4) is

$$\begin{aligned}\mathscr{L}(\mathbf{X},\mathbf{Y},\mathbf{Z};\boldsymbol{\Lambda},\boldsymbol{\Pi}) = tr(\mathbf{ZL}) + \lambda\ell(\mathbf{Z},\mathbf{T}) + \boldsymbol{\Lambda}\bullet(\mathbf{Z}-\mathbf{X}^\top\mathbf{Y}) \\ +\frac{\alpha}{2}\left\|\mathbf{Z}-\mathbf{X}^\top\mathbf{Y}\right\|^2 + \boldsymbol{\Pi}\bullet(\mathbf{X}-\mathbf{Y}) + \frac{\beta}{2}\left\|\mathbf{X}-\mathbf{Y}\right\|^2,\end{aligned}$$

where $\boldsymbol{\Lambda},\boldsymbol{\Pi}$ are the Lagrange multipliers. ADMM then updates the variables as

$$\mathbf{X}^{k+1} = \arg\min_{\mathbf{X}} \mathscr{L}(\mathbf{X},\mathbf{Y}^k,\mathbf{Z}^k;\boldsymbol{\Lambda}^k,\boldsymbol{\Pi}^k), \mathbf{Y}^{k+1} = \arg\min_{\mathbf{Y}} \mathscr{L}(\mathbf{X}^{k+1},\mathbf{Y},\mathbf{Z}^k;\boldsymbol{\Lambda}^k,\boldsymbol{\Pi}^k),$$

$$\mathbf{Z}^{k+1} = \arg\min_{\mathbf{Z}} \mathscr{L}(\mathbf{X}^{k+1},\mathbf{Y}^{k+1},\mathbf{Z};\boldsymbol{\Lambda}^k,\boldsymbol{\Pi}^k),$$

$$\boldsymbol{\Lambda}^{k+1} = \boldsymbol{\Lambda}^k + \alpha(\mathbf{Z}^{k+1}-\mathbf{S}^{k+1}),\ \boldsymbol{\Pi}^{k+1} = \boldsymbol{\Pi}^k + \beta(\mathbf{X}^{k+1}-\mathbf{Y}^{k+1}), \tag{5}$$

where $\mathbf{S}^k \equiv {\mathbf{X}^k}^\top\mathbf{Y}^k$. By straightforward differentiation, the optimization subproblems of $\mathbf{X}^{k+1},\mathbf{Y}^{k+1}$ and $\mathbf{Z}^{k+1}$ can be solved as

$$\mathbf{X}^{k+1} = (\mathbf{A}^k)^{-1}\mathbf{c}^k, \tag{6}$$

where $\mathbf{A}^k = \beta\mathbf{I} + \alpha\mathbf{Y}^k{\mathbf{Y}^k}^\top$, and $\mathbf{c}^k = \mathbf{Y}^k(\boldsymbol{\Lambda}^k + \alpha\mathbf{Z}^k + \beta\mathbf{I})^\top - \boldsymbol{\Pi}^k$;

$$\mathbf{Y}^{k+1} = (\mathbf{B}^k)^{-1}\mathbf{d}^k, \tag{7}$$

where $\mathbf{B}^k = \beta\mathbf{I} + \alpha\mathbf{X}^{k+1}{\mathbf{X}^{k+1}}^\top$, and $\mathbf{d}^k = \mathbf{X}^{k+1}(\boldsymbol{\Lambda}^k + \alpha\mathbf{Z}^k + \beta\mathbf{I}) + \boldsymbol{\Pi}^k$;

$$Z_{ij}^{k+1} = \begin{cases} \min\limits_z f_{ij}(z) & (i,j)\in\Omega \quad (8a) \\ \left(\mathbf{S}^{k+1} - \frac{1}{\alpha}(\boldsymbol{\Lambda}^k + \mathbf{L})\right)_{ij} & (i,j)\notin\Omega, \quad (8b)\end{cases}$$

where $\Omega = \mathcal{S}\cup\mathcal{D}$, and

$$f_{ij}(z) = \lambda\tilde{\ell}(z,T_{ij}) + \frac{\alpha}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2. \tag{9}$$

Hence, the only remaining issue is how to solve (8a).

3.1 Different Loss Functions

In this section, we show that problem (8a) can be easily solved for a variety of loss functions.

ℓ_2-Loss $\tilde{\ell}(z, T_{ij}) = \frac{1}{2}(z - T_{ij})^2$. Problem (8a) can be easily solved by setting the derivative of the objective in (9) to zero, leading to

$$z^* = \frac{1}{\alpha + \lambda}(\alpha S_{ij}^{k+1} + \lambda T_{ij} - \Lambda_{ij}^k - L_{ij}). \tag{10}$$

Hinge Loss $\tilde{\ell}(z, T_{ij}) = \max\{1 - zT_{ij}, 0\}$. Problem (8a) can be rewritten as $\min_{z,\epsilon}\ \lambda\epsilon + \frac{\alpha}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2 :\ T_{ij}z \geq 1 - \epsilon,\ \epsilon \geq 0$. Let θ be the Lagrange multiplier for the constraint $T_{ij}z \geq 1 - \epsilon$. Using the standard method of Lagrange multipliers, it can be easily shown that

$$z^* = \frac{\theta T_{ij} - \Lambda_{ij}^k - L_{ij}}{\alpha} + S_{ij}^{k+1}, \tag{11}$$

where $\theta = \min\left\{\max\left\{\frac{\alpha - T_{ij}(\alpha S_{ij}^{k+1} - \Lambda_{ij}^k - L_{ij})}{T_{ij}^2}, 0\right\}, \lambda\right\}$.

Squared Hinge Loss $\tilde{\ell}(z, T_{ij}) = \frac{1}{2}\max\{1 - zT_{ij}, 0\}^2$. Problem (8a) can be rewritten as $\min_{z,\epsilon}\ \frac{\lambda}{2}\epsilon^2 + \frac{\alpha}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2 :\ T_{ij}z \geq 1 - \epsilon$. Similar to the hinge loss, the optimal solution can be obtained as

$$z^* = \frac{\theta T_{ij} - \Lambda_{ij}^k - L_{ij}}{\alpha} + S_{ij}^{k+1}, \tag{12}$$

where $\theta = \max\left\{\frac{\alpha - T_{ij}(\alpha S_{ij}^{k+1} - \Lambda_{ij}^k - L_{ij})}{\frac{\alpha}{\lambda} + T_{ij}^2}, 0\right\}$.

ℓ_1-Loss $\tilde{\ell}(z, T_{ij}) = |z - T_{ij}|$. With the ℓ_1-loss, $f_{ij}(z)$ in (9) can be written as

$$\begin{aligned} f_{ij}(z) &= \frac{\lambda}{\alpha}|z - T_{ij}| + \frac{1}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2 \\ &= \frac{\lambda}{\alpha}|\hat{z}| + \frac{1}{2}\left(\hat{z} + T_{ij} - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2, \end{aligned}$$

where $\hat{z} = z - T_{ij}$. Hence, problem (8a) becomes a standard problem with ℓ_2-loss and ℓ_1-regularizer, and the optimal $\hat{z}^*$ can be obtained as

$$\hat{z}^* = \mathrm{Th}_{\frac{\lambda}{\alpha}}\left(S_{ij}^{k+1} - T_{ij} - \frac{L_{ij} + \Lambda_{ij}^k}{\alpha}\right),$$

where $\mathrm{Th}_\nu(x) = \begin{cases} x - \nu & x > \nu \\ 0 & -\nu \leq |x| \leq \nu \\ x + \nu & x < -\nu, \end{cases}$ is the soft-thresholding operator. Consequently,

$$z^* = \hat{z}^* + T_{ij} = \mathrm{Th}_{\frac{\lambda}{\alpha}}\left(S_{ij}^{k+1} - T_{ij} - \frac{L_{ij} + \Lambda_{ij}^k}{\alpha}\right) + T_{ij}. \tag{13}$$

3.2 Algorithm

To monitor convergence, we require the primal and dual residuals at iteration $k+1$

$$\Delta_{\text{primal}_1} = \left\|\mathbf{Z}^{k+1} - \mathbf{S}^{k+1}\right\|, \Delta_{\text{primal}_2} = \left\|\mathbf{X}^{k+1} - \mathbf{Y}^{k+1}\right\|, \Delta_{\text{dual}} = \left\|\mathbf{X}^{k+1} - \mathbf{X}^{k}\right\| \tag{14}$$

to be small [3]. Moreover, to improve convergence, it is common to vary the penalty parameters in each ADMM iteration. Specifically, following [3], we update them as

$$\alpha \leftarrow \begin{cases} 2\alpha & \Delta_{\text{primal}_1} > 10\Delta_{\text{dual}} \\ \max(\alpha/2, 1.5) & \Delta_{\text{dual}} > 10\Delta_{\text{primal}_1} \end{cases}, \tag{15}$$

and

$$\beta \leftarrow \begin{cases} 2\beta & \Delta_{\text{primal}_2} > 10\Delta_{\text{dual}} \\ \max(\beta/2, 1.5) & \Delta_{\text{dual}} > 10\Delta_{\text{primal}_2} \end{cases}. \tag{16}$$

The whole procedure is shown in Algorithm 1.

Algorithm 1. Kernel learning by ADMM.

1: **Input:** $\mathbf{X}^0, \mathbf{Y}^0, \mathbf{Z}^0$, parameters ε and $IterMax$.
2: **Output:** $\mathbf{Z} = \mathbf{X}^{k\top}\mathbf{X}^k$ (or $\mathbf{Y}^{k\top}\mathbf{Y}^k$).
3: $k \leftarrow 0$;
4: **repeat**
5: update $\mathbf{X}^{k+1}, \mathbf{Y}^{k+1}$ by (6) and (7) respectively;
6: update $\{Z_{ij}^{k+1} \mid (i,j) \notin \Omega\}$ by (8a);
7: update $\{Z_{ij}^{k+1} \mid (i,j) \in \Omega\}$ by (10), (11), (12) or (13), depending on the loss;
8: update $\boldsymbol{\Lambda}^{k+1}, \boldsymbol{\Pi}^{k+1}$ by (5);
9: update α and β using (15) and (16);
10: compute the primal and dual residuals in (14);
11: $k \leftarrow k+1$;
12: **until** $\max(\Delta_{\text{primal}_1}, \Delta_{\text{primal}_2}, \Delta_{\text{dual}}) < \varepsilon$ or $k > IterMax$.

3.3 Convergence

With the low-rank decomposition, problem (4) is nonconvex w.r.t. $\mathbf{X}$ and $\mathbf{Y}$, and so we can only consider local convergence [3]. As in [13], we show below a necessary condition for local convergence.

Lemma 1. *Let* $\mathbf{W} \equiv (\mathbf{X}, \mathbf{Y}, \mathbf{Z})$, $\boldsymbol{\Gamma} \equiv (\boldsymbol{\Lambda}, \boldsymbol{\Pi})$, *and* $\{\mathbf{W}^k, \boldsymbol{\Gamma}^k)\}$ *be a sequence generated by Algorithm 1. Then,* $\mathscr{L}(\mathbf{W}^k, \cdot) - \mathscr{L}(\mathbf{W}^{k+1}, \cdot) \geq \mu \left\|\mathbf{W}^k - \mathbf{W}^{k+1}\right\|^2$, *and* $\mathscr{L}(\boldsymbol{\Gamma}^k, \cdot) - \mathscr{L}(\boldsymbol{\Gamma}^{k+1}, \cdot) \geq -\frac{1}{\mu}\left\|\mathbf{W}^k - \mathbf{W}^{k+1}\right\|^2$, *where* $\mu = \min\{\alpha, \beta\}$, *and* $\mathscr{L}(\mathbf{X}, \cdot)$ *denotes that all the variables in* $\mathscr{L}$ *except* $\mathbf{X}$ *are fixed.*

Proposition 1. *Let* $\{\mathbf{X}^k, \mathbf{Y}^k, \mathbf{Z}^k, \boldsymbol{\Lambda}^k, \boldsymbol{\Pi}^k\}$ *be a sequence generated by Algorithm 1. If* $\{\boldsymbol{\Lambda}^k, \boldsymbol{\Pi}^k\}$ *is bounded and* $\sum_{k=0}^{\infty}\left(\left\|\boldsymbol{\Lambda}^{k+1} - \boldsymbol{\Lambda}^k\right\|^2 + \left\|\boldsymbol{\Pi}^{k+1} - \boldsymbol{\Pi}^k\right\|^2\right) < \infty$, *then* $\mathbf{X}^k - \mathbf{X}^{k+1} \to \mathbf{0}, Y^k - \mathbf{Y}^{k+1} \to \mathbf{0}, \mathbf{Z}^k - \mathbf{Z}^{k+1} \to \mathbf{0}$, *and any accumulation point of* $\{\mathbf{X}^k, \mathbf{Y}^k, \mathbf{Z}^k\}$ *satisfies the Karush-Kuhn-Tucker condition of (4).*

4 Experiments

As in [5,10], we study the performance of the proposed approach in the context of data clustering. Specifically, a kernel matrix is learned from the pairwise constraints (i.e., *must-link* and *cannot-link*), which is then used for clustering by the kernel k-means algorithm. Experiments are performed on a number of

Table 1. Data sets used in the experiment

data set	#classes	#patterns	#features	#constraints
glass	6	214	9	256
heart	2	270	13	324
iris	3	150	4	180
protein	6	116	20	140
sonar	2	208	60	250
wine	3	178	12	214
odd-even	2	4000	256	4800

Table 2. Comparison on clustering accuracy (%) on the data sets with the squared loss (SL), hinge loss (HL), squared hinge loss (SHL), and ℓ_1-loss (ℓ_1). The best and comparable results (according to the pairwise t-test with 95% confidence) are highlighted.

loss	method	glass	heart	iris	protein	sonar	wine	odd-even
SL	ADMM	**80.9±1.1**	**93.3±1.9**	**99.0±1.1**	**87.7±1.9**	**95.5±2.5**	**83.2±1.8**	**99.37±0.06**
	BCD	**80.8±1.0**	**93.4±2.5**	**99.2±0.9**	86.0±1.9	**95.7±2.5**	82.9±1.7	**99.57±0.03**
	S-NPKL	80.3±1.8	91.7±2.5	**99.2±1.0**	86.7±2.5	**95.2±2.6**	82.9±1.8	**99.35±0.09**
HL	ADMM	**80.1±1.2**	**92.4±2.5**	**99.2±1.0**	**87.2±1.7**	**95.9±2.4**	**83.4±1.6**	**99.35±0.05**
	BCD	79.6±1.9	85.9±3.0	98.9±1.0	85.9±2.0	94.2±3.9	**83.2±1.8**	**99.55±0.05**
	S-NPKL	**80.6±1.1**	88.1±4.1	**99.1±0.8**	85.0±2.5	**95.5±2.4**	**83.3±1.6**	**99.42±0.08**
	SDPLR	79.6±1.5	90.0±2.8	98.9±0.7	83.1±2.8	94.9±1.2	82.1±2.3	98.35±0.20
SHL	ADMM	80.7±1.6	**93.4±2.0**	**99.0±1.1**	**86.8±1.8**	**95.5±2.5**	**83.1±2.1**	**99.35±0.10**
	BCD	**81.0±1.6**	**93.4±2.5**	**99.2±0.9**	**86.4±2.3**	**95.7±2.5**	**82.9±1.7**	**99.57±0.03**
	S-NPKL	80.5±2.0	**93.2±2.2**	**99.2±1.0**	82.0±2.2	**95.5±2.5**	**83.0±2.0**	**99.35±0.09**
ℓ_1	ADMM	79.1±1.8	88.0±4.5	98.2±1.6	84.4±2.5	94.7±2.4	80.5±2.4	96.62±0.12

Table 3. Comparison on CPU time (second) on the data setswith the squared loss (SL), hinge loss (HL), squared hinge loss (SHL), and ℓ_1-loss (ℓ_1). The best and comparable results (according to the pairwise t-test with 95% confidence) are highlighted.

loss	method	glass	heart	iris	protein	sonar	wine	odd-even
SL	ADMM	**5.5±1.2**	**10.4±2.7**	2.3±1.0	**1.6±1.6**	**6.8±1.8**	**3.6±2.3**	**359±12.5**
	BCD	9.9±4.2	28.9±10.1	**0.9±0.1**	2.2±1.5	8.3±4.9	4.0±2.1	653±18.4
	S-NPKL	46.2±19.3	311.0±149.2	23.4±7.6	60.8±40.2	176.0±58.7	37.0±17.5	3,680±219.8
HL	ADMM	**6.9±1.8**	**12.8±7.8**	**2.9±1.5**	**1.8±1.5**	**7.4±3.0**	**4.9±2.2**	**361±10.4**
	BCD	163.6±49.7	319.7±44.8	34.8±26.8	41.8±16.9	217.8±45.3	81.8±35.5	5,694±184.2
	S-NPKL	59.8±13.5	224.0±106.2	41.9±6.5	44.0±8.06	180.7±82.7	49.1±23.1	1,955±352.2
	SDPLR	17.4±6.6	40.1±11.2	8.8±2.0	4.7±0.9	21.6±9.8	9.4±5.7	20,065±374.6
SHL	ADMM	**5.1±1.4**	**11.5±5.4**	**1.9±1.2**	**1.2±0.6**	**5.1±1.7**	**3.9±2.4**	**357±11.1**
	BCD	78.3±42.1	231.9±72.8	5.5±2.9	27.2±25.1	56.8±43.8	44.8±21.3	6,030±237.7
	S-NPKL	57.76±17.6	157.0±86.0	15.3±3.1	43.6±8.9	131.2±56.3	55.4±29.5	1,736±284.4
ℓ_1	ADMM	7.7±1.8	13.6±5.8	3.6±1.8	1.7±0.3	8.4±1.7	5.5±2.2	413±8.2

benchmark data sets[2] (Table 1) that have been commonly used for nonparametric kernel learning [5,6,14].

The proposed *Algorithm 1* (denoted "ADMM") is compared with the three solvers: block coordinate descent method (denoted "BCD") [6], simple nonparametric kernel learning (denoted "S-NPKL") [14] and low-rank SDP (denoted "SDPLR") [4]. Similar to ADMM, all are based on a rank-r approximation of $\mathbf{Z}$. Moreover, we follow [14,6] and set the rank of the kernel matrix to the largest r satisfying $r(r+1)/2 \leq m$, where m is the total number of constraints in Ω.

Results on the clustering accuracy and CPU time are shown in Tables 2 and 3, respectively. As can be seen, ADMM is more efficient than the other methods, while yielding comparable clustering accuracy.

5 Conclusion

In this paper, we proposed an efficient solver for nonparametric low-rank kernel learning. Using ADMM, it decouples the optimization problem into computationally inexpensive subproblems that involve only individual entries of the kernel matrix. Moreover, with an explicit low-rank factorization, it no longer needs to enforce the psd constraint that would lead to expensive eigen-decomposition in each iteration. Experimental results on a number of real-world data sets demonstrate that the proposed method is as accurate as directly solving the SDP, but is much faster than existing solvers.

Acknowledgment. This research was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant No. 614012), the National Natural Science Foundation of China (No. 61165012) , the Natural Science Foundations Fund (Grant No. 2011FZ074) and the Department of Education Fund (No. 2011Y305) of Yunnan Province.

References

1. Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the SMO algorithm. In: Proceedings of the Twenty-First International Conference on Machine, Banff, Alberta, Canada, pp. 6–13 (July 2004)
2. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research 7, 2399–2434 (2006)
3. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 1–122 (2011)
4. Burer, S., Choi, C.: Computational enhancements in low-rank semidefinite programming. Optimization Methods and Software 21(3), 493–512 (2006)

[2] Downloaded from the UCI repository (`http://archive.ics.uci.edu/ml/`).

5. Hoi, S.C.H., Jin, R., Lyu, M.R.: Learning nonparametric kernel matrices from pairwise constraints. In: Proceedings of the Twenty-Fourth International Conference on Machine Learning, Corvallis, Oregon, USA, pp. 361–368 (June 2007)
6. Hu, E.-L., Wang, B., Chen, S.: BCDNPKL: Scalable non-parametric kernel learning using block coordinate descent. In: Proceedings of the Twenty-Eighth International Conference on Machine Learning, Bellevue, WA, USA, pp. 209–216 (June 2011)
7. Kulis, B., Sustik, M., Dhillon, I.: Low-rank kernel learning with Bregman matrix divergences. Journal of Machine Learning Research 10, 341–376 (2009)
8. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., El Ghaoui, L., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. Journal of Machine Learning Research 5, 27–72 (2004)
9. Li, Z., Liu, J.: Constrained clustering by spectral regularization. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Miami, Florida, USA, pp. 421–428 (June 2009)
10. Li, Z., Liu, J., Tang, X.: Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In: Proceedings of the Twenty-Fifth International Conference on Machine Learning, Helsinki, Finland, pp. 576–583 (July 2008)
11. Shang, F., Jiao, L.C., Wang, F.: Semi-supervised learning with mixed knowledge information. In: Proceedings of the Eighteenth Conference on Knowledge Discovery and Data Mining, Beijing, China, pp. 732–740 (August 2012)
12. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained K-means clustering with background knowledge. In: Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA, USA, pp. 577–584 (June 2001)
13. Xu, Y., Yin, W., Wen, Z., Zhang, Y.: An alternating direction algorithm for matrix completion with nonnegative factors. Technical Report TR11-03, Department of Computational and Applied Mathematics, Rice University (2011)
14. Zhuang, J., Tsang, I.W., Hoi, S.C.H.: A family of simple non-parametric kernel learning algorithms. Journal of Machine Learning Research 12, 1313–1347 (2011)

Topic Model Kernel: An Empirical Study towards Probabilistically Reduced Features for Classification

Tien-Vu Nguyen, Dinh Phung, and Svetha Venkatesh

Centre for Pattern Recognition and Data Analytics (PRaDA)
Deakin University, Australia
{tvnguye,dinh.phung,svetha.venkatesh}@deakin.edu.au

Abstract. Probabilistic topic models have become a standard in modern machine learning with wide applications in organizing and summarizing 'documents' in high-dimensional data such as images, videos, texts, gene expression data, and so on. Representing data by dimensional reduction of mixture proportion extracted from topic models is not only richer in semantics than bag-of-word interpretation, but also more informative for classification tasks. This paper describes the Topic Model Kernel (TMK), a high dimensional mapping for Support Vector Machine classification of data generated from probabilistic topic models. The applicability of our proposed kernel is demonstrated in several classification tasks from real world datasets. We outperform existing kernels on the distributional features and give the comparative results on non-probabilistic data types.

Keywords: Classification, Kernel Method, Support Vector Machine, Topic Modelling Features, Latent Dirichlet Allocation, Hierarchical Dirichlet Process.

1 Introduction

Data representation is critical in data analysis tasks. Central to Support Vector Machines are kernels, that maps the input data to another dimensional spaces in which the linear separating hyperplanes are easier to construct. The choice of kernel is crucial . In this paper we focus on a class of problem for SVM when the data can be conveniently represented in distributional forms. Such distributions constitute rich information one can exploit, as they are outputs from the probabilistic topic models [1], latent variables can be used as distributional representation for data. Examples include Latent Dirichlet Allocation [1] or Hierarchical Dirichlet Processes [2], that can produce multinomial distributions over topics given text data or raw pixels in images. This distribution is not only richer in semantics than the original bag of words, but also Blei et al [1] have verified that the reduced representation by topic model features attains richer information for classification than the original word feature as treating individual words as feature. Moreover, such derived features occupy only 99.6 percent in space compared to a very large raw feature set of individual words.

The combinations of generative approaches (such as LDA, HDP) with discriminative ones (e.g. SVM) have recently shown to be very effective [3], hence it is attractive to expose methods that integrate these statistical models and discriminative classifiers.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 124–131, 2013.

Furthermore, we are motivated by recent successful applications of Jensen Shannon divergences to compute the similarities and distances when the data are drawn from probabilistic distributions [4,5].

We make use of preprocessing raw data with topic models, allows exploitation of powerful, latent semantics. Together with the use of divergence in probability space, our main contributions are derivation of a proper kernel originated from the Jensen-Shannon divergence [6], which we call Topic Model Kernel (TMK). The fact that recent advance in topic modelling and Bayesian nonparametrics, such as the HDP [2] which automatically determine the number of topics make the proposed framework more attractive to real-world application. We conducted extensive experimental validation of the proposed TMK which outperforms other existing kernels on the probabilistic derived feature and yields a comparative performance on other data types (free-distribution guarantee).

2 Related Background

2.1 Support Vector Machines and Kernel Method

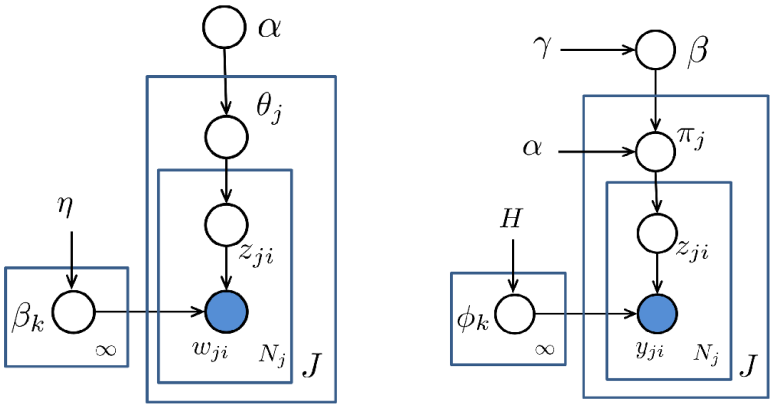
Support Vector Machines (SVM) [7] is a well-known supervised learning method for classification . The goal is to find a better representation by mapping the data into a high dimensional feature space, a fundamental step in SVM. Because the mapping can be general, there are numerous existing kernels in literature[1]. Each kernel is taking into account different 'genres' of the real world data type for the best performance. The most appropriate kernel must guarantee the smoothness amongst data within the same class, maintain distinction to others classes. We choose the four baseline kernels for comparison built-in in LibSVM [8]:

- Radial Basic Function Kernel (RBF) $k(x,y) = \exp\left(-\gamma \parallel x-y \parallel^2\right)$, the parameter γ plays a crucial role in the classification performance.
- Linear Kernel $k(x,y) = x^T y + c$, where c is a constant.
- Polynomial Kernel $k(x,y) = \left(\alpha x^T y + c\right)^d$, with polynomial degree d.
- Sigmoid Kernel $k(x,y) = \tanh\left(\alpha x^T y + c\right)$ where slope parameter α need to be adjusted for best performance.

2.2 Probabilistic Topic Models

The discrete distribution features in practice can be the outcome of probabilistic topic models. Blei et al [1] introduce Latent Dirichlet Allocation (LDA) which is a class of topic modelling that provides a simple way to analyze large volumes of unlabeled text. At the first glance, LDA can be seen as mixture distribution, comprising an underlying set of distributions interpreting the complex data into a group of simpler densities. A 'topic' consists of the group of words that frequently occur together. There are K topics β_k, $k \in \{1, ..., K\}$ which are discrete distributions over words. Then, each document is assumed to be characterized by a mixture proportion (the latent variable θ_j and π_j on Figure 1). This document feature representation is a k-dimensional vector where an element k-th indicates how much the document j contributes to the topic k-th.

[1] Many kernels can be found in `http://crsouza.blogspot.com.au/2010/03/kernel-functions-for-machine-learning.html`



(a) LDA Representation (b) HDP Representation

Fig. 1. Probabilistic Topic Models

Two noticeable versions of topic modelling in parametric and nonparametric settings are Latent Dirichlet Allocation [1] and Hierarchical Dirichlet Processes [2]. Due to limited space, the details of probabilistic topic models can refer to our technical report [9].

3 Topic Model Kernel

3.1 Kullback–Leibler Divergence

The Kullback–Leibler (KL) divergence [10] is a non-symmetric measure of the similarity between two probability distributions. Its intuitive understanding arises from likelihood theory measuring the distance between the initialized probability parameter. The KL divergence from distribution P to Q for discrete case is defined as:

$$D_{KL}\left(P \parallel Q\right) = \sum_i P(i) \ln \frac{P(i)}{Q(i)} \tag{1}$$

where p and q denote the densities of the distributions P and Q. Moreno et al [11] have proposed a symmetric KL divergence kernel for classifying objects under the Gaussian mixture models, a step toward classifying distribution data with SVM.

3.2 Jensen–Shannon Divergence

Based on the KL divergence, the Jensen-Shannon (JS) divergence [6] calculates the distance between two probability distributions P and Q as:

$$D_{JS}\left(P, Q\right) = \pi D_{KL}\left(P \parallel M\right) + \left(1 - \pi\right) D_{KL}\left(Q \parallel M\right) \tag{2}$$

where $M = \frac{1}{2}\left(P + Q\right)$, and D_{KL} is the KL divergence. The lower bound of JS divergence is 0 when two distributions are identical. Its square root [6] is proved as a metric with the triangle inequality property for two distributions. This distance can be seen (in the symmetric KL flavour) as the average distance between two random distributions to their empirical mean, with π is set as 0.5 [12].

3.3 Topic Model Kernel

The kernel function is basically a measurement criteria that compares the similarity between two points or vectors. But not all of the measurement distances or similarity functions yield proper attributes to be a valid kernel. The Topic Model Kernel (TMK) is defined as following:

$$K_{TM}(X,Y) = \exp\left\{-\frac{1}{\sigma^2} \times D_{JS}(X,Y)\right\}$$
$$= \exp\left\{-\frac{1}{\sigma^2} \times \left[\frac{1}{2}\sum_i X(i) \ln \frac{X(i)}{M(i)} + \frac{1}{2}\sum_i Y(i) \ln \frac{Y(i)}{M(i)}\right]\right\} \quad (3)$$

By exponentiating the negative JS divergence, it leads to the positive definite kernel function K_{TM} because (1) JS divergence is negative definite on $R_+ \times R_+$ [13], (2) let exponentiate the negative of JS divergence giving the positive definite kernel that projecting the divergence distance into the bounded range of 0 and 1. Thus, TMK satisfies the Mercer condition of $c^T K_{TM} c \geq 0$ with $K_{TM(i,j)} = k_{TM}(x_i, x_j)$ for the validity of the kernel. The variance σ^2 plays a role as a shape parameter to more flexibly flatting or widening the data.

4 Experiments

Experiments are conducted using real world data in numerous classification scenarios, including: (1) the topic model features derived from LDA and HDP inference, (2) the raw features that we do not guarantee them fit into any type of distribution (3) we further analyze the kernel performance on parameter space. LibSVM [8] is used as a standard library to compare the proposed kernel with four baseline LibSVM built-in kernels mentioned in section 2.1. The data will be scaled as recommended in LibSVM to ensure the best performance.

The scores are reported in two manners at the default parameter (set by LibSVM) and the optimal parameter (for the best performance) by brute-force cross validation searching. For Topic Model Kernel, we empirically set the value of our default parameter σ^2 is equal to the feature dimension size after observing TMK operations on several datasets. Each training and testing set are randomly selected and run SVM for 10 times.

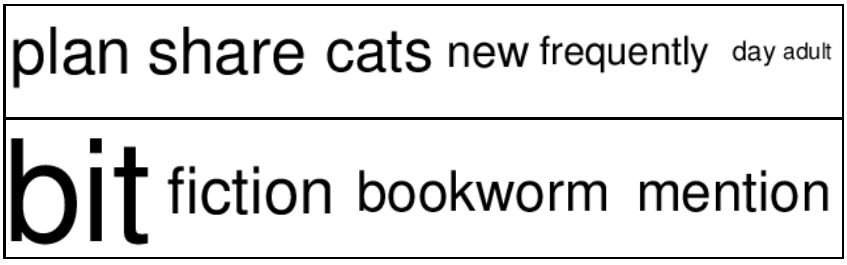


Fig. 2. Two examples of LDA topic β_k on LiveJournal Data

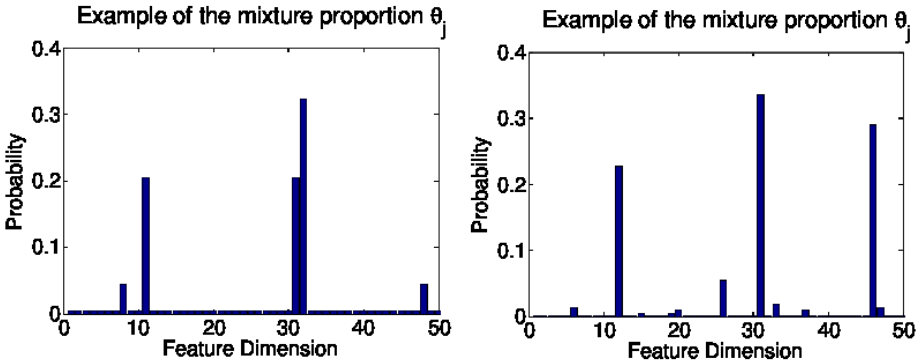


Fig. 3. Examples of the reduced feature θ_j by LDA from 65,483 to 50

4.1 Topic Model Feature

We run LDA and HDP to extract the mixture proportions θ_j (LDA) and π_j (HDP), then use SVM for classification on LiveJournal, Reuter21578, and LabelMe dataset.

4.1.1 LDA Feature on LiveJournal and Reuter21578 Datasets

We crawled the communities listed in the Livejournal directory[2]. These communities are categorised by Livejournal into 10 groups , summarizing of 8,758 posts giving the vocabulary size of 65,483 (the feature dimension of raw data). The task is to predict the category, given text data from user's posts.

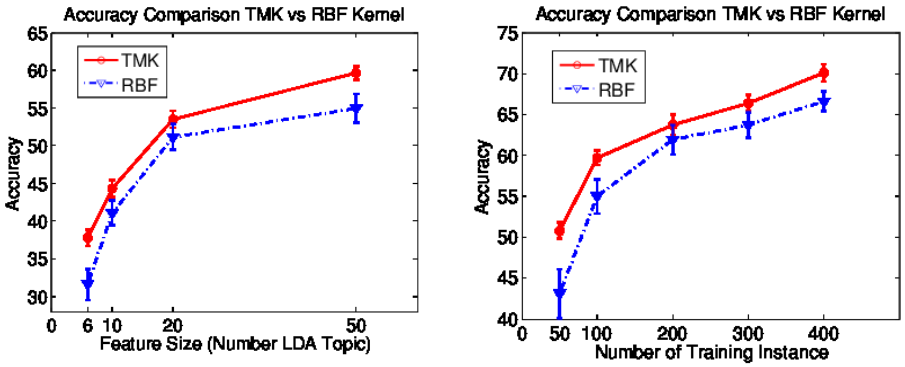
We treat each user post as a document and run LDA with fixed number of latent factors from $\{6, 10, 20, 50\}$.The examples of estimated topic β_k, about literature and life, are visualized in Figure 2 and the LDA feature examples are in Figure 3 which reduced from original high dimension of 65483 to 50. We do the experiments progressively with increasing numbers of training samples from 10 to 400 (refer Figure 4b and varying the number of hidden factors K (refer Figure 4a) to clearly shows the effect of increasing the number of learned feature or number of training instance.

Similar to Live Journal data, we utilize posterior inference of LDA on Reuters21578 dataset to extract the mixing proportion feature θ_j in which the number of hidden factors is set as 20. The results in Figure 4 and Table 5 demonstrate the superiority of our kernel.

4.1.2 HDP Feature on LabelMe Dataset

LabelMe [14] is one of the most well known benchmark dataset for image annotations and object categorizations. To discard the noise and mistagging issues, top 30 high frequency tags are chosen giving a vocabulary size of 30. The Hierarchical Dirichlet Processes [2] is carried out to extract the topic assignment feature properly, each image is treated as a document while each tag is considered as word y_{ji} (refer 1b) in the model. HDP automatically identifies 24 topics ϕ_k giving the feature π_j dimension of 24.

[2] `http://www.livejournal.com/browse/`, retrieved August 2012.



(a) With varied feature dimension.

(b) With varied number of training data.

Fig. 4. Performances comparison on LDA feature derived from LiveJournal

	Dataset\Kernel	TMK	RBF	Linear	Polynomial	Sigmoid
Default Parameter	LiveJournal	**58.1**±2.1	54.9±4.9	54.4±5.2	52.6±6.6	51.8±5.1
	Reuter21578	**81.3**±0.2	79.0±0.5	78.2±0.1	77.9±0.1	77.4±0.4
	LabelMe	**72.3**±**1.9**	70.8±1.9	71.5±1.8	62.7±4.2	69.8±1.6
	MNIST	88.4±0.9	82.2±1.9	**91.3**±**0.5**	83.7±2.6	79.**6**±3.4
Optimal Parameter	LiveJournal	**58.7**±1.7	55.0±4.8	54.9±4.2	54.2±5.2	53.5±4.7
	Reuter21578	**81.9**±0.1	79.4±0.5	79.0±0.1	78.9 + 0.1	79.2±0.3
	LabelMe	**76.1**±1.8	73.3±2.0	74.8±1.4	74.5±2.2	73.9±1.3
	MNIST	**90.8**±2.2	89.3±2.5	88.4±2.7	88.7±2.9	85.8±2.7

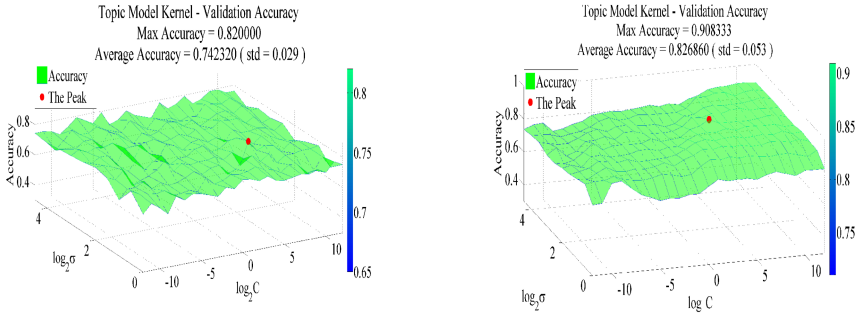
Fig. 5. SVM classification comparison with TMK and four baseline kernels

4.2 Non-distributional Data Source

To highlight the generality, we show how the proposed kernel performs on the raw (non-distribution) data of MNIST dataset instead of extracting topic model features as previously.

MNIST Dataset. The ready-to-use extracted feature in MNIST dataset, a benchmark in handwritten digit recognition, is available at author website [3] . In this experiment, we do not aim to beat the state of the art result on MNIST, but we want to illustrate the classification comparison between the TMK versus others with SVM tool. We note that this kind of raw image data is not pledged to be drawn from any type distribution when use with TMK for classification. The accuracy of 600 training with 100 testing instances is recorded in Table 5, although Linear kernel perform very well with default parameter, our kernel achieves the best result with an optimal parameter (after parameter selection).

[3] `http://yann.lecun.com/exdb/mnist/`



(a) HDP feature on LabelMe dataset. (b) MNIST dataset.

Fig. 6. Cross validation accuracy by brute-force parameter searching

4.3 Parameter Space Analysis

We now move on to our characterization of performance on various axes of parameters. To demonstrate the TMK kernel is more robust on the parameter space, we record the accuracy planes with parameter C in SVM optimization equation [8] and TMK parameter σ shown in Figure 6a. We get the peak accuracy of 0.82 on by 3 fold cross validation at which the optimal parameter is further used for testing. The average accuracy with standard deviation is used to evaluate the preeminent of TMK when the data is drawn from distribution. Topic Model Kernel accomplishes the best in the way that it get the highest score on average accuracy (0.74), lowest standard deviation (0.029), and the TMK's peak (0.82) is the highest among four baseline kernel's peaks (refer Table 1).

Table 1. Accuracy on parameter space comparison

Kernels	LabelMe: HDP probabilistic feature			MNIST: Non-probabilistic feature		
	Peak	Average	Std	Peak	Average	Std
TMK	**0.82**	**0.74**	**0.029**	**0.91**	**0.83**	0.053
RBF	0.77	0.70	0.033	0.90	0.67	0.252
Linear	0.75	0.70	0.034	0.88	0.80	**0.029**
Polynomial	0.75	0.35	0.236	0.88	0.44	0.295
Sigmoid	0.76	0.69	0.041	0.85	0.43	0.304

Further, we would like to see the performance of TMK on the non-distribution feature by varying the parameters of TMK. Although, it is not really stable as on topic model features, at a certain area (obtained by cross validation) it perform pretty well with comparable accuracy to others kernel.

5 Conclusion

We introduced the Topic Model Kernel, a bridge connecting between the information theory and kernel method, and compared it to other existing kernels in SVM

classification tasks. The significant applications of this work in practical data are examined on the influential feature derived from recent probabilistic topic frameworks of LDA and HDP that the TMK outperforms other existing kernels on topic model feature (drawn from probabilistic assumption). We also show its comparative performance on the natural application of digit recognition (without any assumption of distribution).

References

1. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
2. Teh, Y., Jordan, M., Beal, M., Blei, D.: Hierarchical Dirichlet processes. Journal of the American Statistical Association 101, 1566–1581 (2006)
3. Fritz, M., Schiele, B.: Decomposition, discovery and detection of visual categories using topic models. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
4. Antolín, J., Angulo, J., López-Rosa, S.: Fisher and jensen–shannon divergences: Quantitative comparisons among distributions application to position and momentum atomic densities. The Journal of Chemical Physics 130, 074110 (2009)
5. Nguyen, T., Phung, D., Gupta, S., Venkatesh, S.: Extraction of latent patterns and contexts from social honest signals using hierarchical dirichlet processes. In: IEEE International Conference on Pervasive Computing and Communications, PerCom 2013 (2013)
6. Endres, D., Schindelin, J.: A new metric for probability distributions. IEEE Transactions on Information Theory 49, 1858–1860 (2003)
7. Boser, B., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers, 144–152 (1992)
8. Chang, C., Lin, C.: Libsvm: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2, 27 (2011)
9. Nguyen, T.V., Phung, D., Venkatesh, S.: Topic model kernel: An empirical study towards probabilistically reduced features for classification. Technical report, Pattern Recognition and Data Analytics, Deakin University (2013)
10. Kullback, S., Leibler, R.: On information and sufficiency. The Annals of Mathematical Statistics 22, 79–86 (1951)
11. Moreno, P.J., Ho, P., Vasconcelos, N.: A kullback-leibler divergence based kernel for svm classification in multimedia applications. Advances in Neural Information Processing Systems 16, 1385–1393 (2003)
12. Chan, A.B., Vasconcelos, N., Moreno, P.J.: A family of probabilistic kernels based on information divergence. Univ. California, San Diego, CA, Tech. Rep. SVCL-TR-2004-1 (2004)
13. Topsoe, F.: Jenson-shannon divergence and norm-based measures of discrimination and variation (2003) (preprint)
14. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. International Journal of Computer Vision 42, 145–175 (2001)

Training Neural Networks with Implicit Variance

Justin Bayer, Christian Osendorfer,
Sebastian Urban, and Patrick van der Smagt

Technische Universität München, Fakultät für Informatik, Lehrstuhl für Robotik und Echtzeitsysteme, Boltzmannstraße 3, 85748 München
bayer.justin@googlemail.com, osendorf@in.tum.de, {surban,smagt}@tum.de

Abstract. We present a novel method to train predictive Gaussian distributions $p(z|x)$ for regression problems with neural networks. While most approaches either ignore or explicitly model the variance as another response variable, it is trained implicitly in our case. Establishing stochasticty by the injection of noise into the input and hidden units, the outputs are approximated with a Gaussian distribution by the forward propagation method introduced for fast dropout [1]. We have designed our method to respect that probabilistic interpretation of the output units in the loss function. The method is evaluated on a synthetic and a inverse robot dynamics task, yielding superior performance to plain neural networks, Gaussian processes and LWPR in terms of likelihood.

Keywords: neural networks, predictive distributions, deep learning.

1 Introduction

Deep learning stands at the center of several key advancements in visual and audio recognition in the past few years [2,3]. The stacking of several layers of computation results in a hierarchy of feature detectors of which each conveys new intermediate representations of the data. Prediction is assumed to be substantially easier in these representational spaces than in input space, since previously entangled "factors of variation" [4] are well separated. Initially, research on deep learning was started by [5] of which a crucial ingredient was the greedy layer wise pretraining in an unsupervised fashion. Yet, the increase in availabilty of computational power (especially in the form of GPUs) showed that deep neural networks can as well be trained if lots of data is available [6], special non-saturating units are used [7] or with the help of a powerful regularizer named dropout [8]. The latter randomly discards units in the network during training, making co-adaption of units and thus poor generalization less likely.

Perceiving the units of a network as stochastic entities goes back at least to [9]. While neural networks can be used to represent any output distribution which can be summarized by a finite number of sufficient statistics, these have to be defined a priori and are not part of the learning. Contrary, the promise of

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 132–139, 2013.

stochastic networks is that $p(z|x)$ can have a number of maxima exponential in the number of units, allowing complex one-to-many relations.

Research on these models has seen notable papers most recently. For one, [10] introduces a deep density estimator $p(x)$ which can be efficiently trained via back-propagation [11] and is a consistent estimator of the underlying data distribution. Learning a stochastic feed-forward net leading to a multi-modal output distribution $p(z|x)$ is made practical in [12]. Novel techniques to train stochastic neural networks with back-propagation have been presented in [13].

Using deep architectures for the estimation of predictive distributions has been tackled before in [14,15]. Our contribution is to make use of findings from [1] to approximate each unit in a stochastic neural network up to second order with a Gaussian and reflect this in the construction of the loss. This leads to unimodal Gaussian predictive distributions which play nicely with dropout regularization for deep neural networks.

2 Approach

The well known method of mixture density networks [16] and a recent development in approximating dropout [1] lie at the heart of our work. A brief overview of both will be given in order to establish a base upon which our contribution can be described. We begin with a short review of neural networks.

2.1 Neural Networks

Neural networks can be described as a stack of *layers* of which each consists of an adaptable affine transformation and a subsequent nonlinear function. Let $x \in \mathbb{R}^I$ be an input to the network from which we wish to produce an output $y \in \mathbb{R}^O$. Given a network of K layers, we compute the output u of a layer given the output of the previous layer u' via the following equation

$$u = f(u'W + b), . \tag{1}$$

The weight matrix W, the bias term b and the transfer function f are layer specific. The whole set of adaptable parameters is referred to as $\theta = \{(W^k, b^k)\}_{k=1}^K$ where we added the top index to distinguish between parameters from different layers. Typical choices for the transfer functions $\{f^k\}_{k=1}^K$ are the *sigmoid* $f(\kappa) = \frac{1}{1+\exp(-\kappa)}$, the *rectifier* $f(\kappa) = \max(\kappa, 0)$ or the identity $f(\kappa) = \kappa$, where $\kappa \in \mathbb{R}$. Transfer functions are applied component wise. A single component of a layer is referred to as a unit or neuron.

The most popular and arguably most efficient way to adapt the behaviour as desired is backpropagation [11,17,18]. Doing so involves the definition of a loss function $\mathcal{L}(\theta)$ which can be differentiated with respect to the parameters θ and fed into an optimizer such as stochastic gradient descent or nonlinear conjugate

gradient. One of the most common choices for a loss function is the mean squared error:

$$\mathcal{L}_{\text{MSE}}(\theta) = \frac{1}{N}\sum_{i}^{N} ||y_i - z_i||_2^2. \tag{2}$$

It is defined as a sum over a data set $\mathcal{D} = \{(x_i, z_i)\}_{i=1}^N$ which consists of independent samples from a function which the network shall mimic. A probabilistic interpretation is that the network models the data with a Gaussian conditioned on the input: $p(z|x) = \mathcal{N}(\mu(x), \sigma^2)$, where the mean is defined by the output of the network, i.e. $\mu(x) = y$. The variance σ^2 is assumed to be constant, which is commonly referred to as *homoscedastic* variance. Taking the log of the likelihood of the data $\prod_i p(z_i|x_i)$ and neglecting constant terms irrelevant to optimization leaves us with Equation (2).

2.2 Density Networks

Mixture density networks [16] are ordinary neural networks with special transfer function at the last layer and a matching loss. The output represents the sufficient statistics of a mixture of Gaussians: priors, means and covariances of each component. The resulting model can be trained via maximum likelihood by numerical minimization of the negative log-likelihood. We consider mixture density networks with only a single component and a diagonal covariance, and thus call them density networks for brevity. Given a regression problem of target dimensionality D, that is $\mathcal{D} = \{(x_i, z_i)\}$ with $z_i \in \mathbb{R}^D$, the output layer is designed to be of size $O = 2D$. The first D components represent the mean while the second D give the variance of a Gaussian:

$$\mu(x_{i,d}) = y_{i,d}, \tag{3}$$
$$\sigma^2(x_{i,d}) = y_{i,D+d}^2. \tag{4}$$

The square assures positive variance. We use these values to specify a conditional Gaussian distribution of $p(z|x) = \mathcal{N}(\mu(x), \sigma^2(x))$ which is *heteroscedastic* since the variance depends on the input. Computing and differentiating the log-likelihood of the targets given the inputs is now straightforward and leads to training via the average negative log-likelihood:

$$\mathcal{L}_{NLL}(\theta) = \frac{1}{N}\sum_{i,d} \frac{(z_{i,d} - \mu(x_{i,d}))^2}{2\sigma^2(x_{i,d})} + \log\sqrt{2\pi\sigma^2(x_{i,d})}. \tag{5}$$

In practice, optimization can be difficult: the variances might collapse to very small numbers, leading to very high likelihoods and numerical instabilities [19]. As a counter measure we added a constant term of 0.0001 to the variances in the objective function and its gradients.

2.3 Fast Dropout

Dropout [8] is a powerful regularizer for neural networks. By randomly neglecting input and hidden units from the network during forward- and back-propagation, the method prevents different units of the network from coadapting and subsequently depending on each other too much. This results in a vast improvement of performance and lead to several significant improvements in visual and audio recognition.

A problem of dropout is that training times tend to be rather long due to the necessesity of sampling and the resulting noisy gradients. To circumvent this, the author's of [1] proposed to approximate the input to each layer of the net $a = u'W + b$. Using a diagonal Gaussian $\hat{a} \sim \mathcal{N}(\mu, s^2)$ is reasonable due to the central limit theorem. Given some mild conditions on the distribution of u' as well as its mean ν' and variance τ'^2 the first two moments of $\hat{a}$ can be computed exactly:

$$m = d(\nu' W + b), \tag{6}$$

$$s^2 = \mathrm{diag}(d(1-d)\nu'^2 W^2 + d\tau'^2 W^2). \tag{7}$$

Here, d refers to the dropout rate. Obtaining the moments ν and τ^2 of $o = f(\hat{a})$ can then be done by propagating μ and s^2 through f. This is possible in closed form for the rectifing linear and approximately for the sigmoid. In other cases, the unscented transform [20] or sampling can be used. The authors of [1] provide more details; no significant loss in performance and yet significant improvements in training time is reported, due to less sampling operations.

2.4 Implicit Variance Networks

A consequence of treating each unit in a network as stochastic and approximating it up to second order is that the output of the network is also stochastic. Since fast dropout already handles the forward propagation of variance, in contrast to plain neural networks, the variance of the output units is readily available. We propose to not neglect the variance at the output layer. Instead we incorporate it into the negative log-likelihood (Equation (5)), where we model the statistics with the output of the last layer, i.e. $\mu(x_i) = m_i^K$ and $\sigma^2(x_i) = (s_i^K)^2$.

We extended the model further in two ways: incorporating input variance and a special bias for the variance of units. Variance of inputs arises naturally in many settings (e.g. noisy sensor data) and can be respected during forward propagation from the inputs to the first hidden layer. A principled treatment for inferring the variance of the inputs can be employed, but we assume it to be constant over the data set and dimensions, essentially treating it as another hyper parameter.

During preliminary experiments, we noticed that the network duplicates units at the last hidden layer to reduce variance at the output layer where required for further minimizing the loss. This effect, which we call "pseudo pruning" is due to the (invalid) assumption of independency between the activations of a layer. The network can just copy a unit and half their outgoing weights to reduce the variance contribution while maintaining the mean contribution to the following layer.[1]

[1] We omit a formal argument due to space restrictions.

While pseudo pruning might seem desirable as it reduces overfitting, it can lead to extreme underfitting and is also computationally not efficient. The forward propagation of variance (given in Equation (7)) was thus adapted to the following:

$$s^2 = \mathrm{diag}(d(1-d)\nu'^2 W^2 + d\tau'^2 W^2) \odot \beta, \tag{8}$$

where $\odot$ is the element wise product and $\beta > B$ is constrained to be greater than some positive number B; this can be done by elegantly by reparametrizing $\beta = \exp(\hat{\beta}) + B$ and optimizing with respect to $\hat{\beta}$. B is treated as a hyper parameter.

3 Experiments

We present experiments on a synthetic data set involving heteroscedastic behaviour as a sanity check for our model. We then move to a real world benchmark where inverse robot dynamics are to be learned; this task has been tackled previously in [21,22]. [2]

For all experiments, the inputs as well as the targets were normalized to zero mean and unit variance; the former helps with optimization [23] and the latter leads to more comparable results for evaluation as it is equivalent to using the normalized MSE. Determination of good hyper parameters was performed via a random search as recommended by [24] for which the search distribution is given in Table 2; batch size and number of hidden units are data set specific and given in the respective section. We picked 32 random configurations for each experiment. Training took place for a fixed number of epochs after which the network with the best score on a held out validation set was picked for final evaluation on the test data. We minimized the $\mathcal{L}_{\mathrm{MSE}}$ in case of NN and FD and $\mathcal{L}_{\mathrm{NLL}}$ for DN and IVN. Optimization was performed with rmsprop [25] using Nesterov momentum [26,27]. For the plain neural networks, where no variance is modelled, we assume homoscedastic variance which we estimate after training as the variance of the residuals. We report the mean squared error and the negative log likelihood, both averaged over the test sets, in Table 1.

3.1 Toy Data

This data set was proposed by [28] in order to evaluate the ability of a model to express heteroscedasticity in an easily inspectable way. It is governed by the following two equations,

$$\mu_i = 2(\exp(-30(x_i - \frac{1}{4})^2) + \sin(\pi x_i^2)),$$
$$\sigma_i^2 = \exp(\sin(2\pi x_i)),$$

[2] We also considered the "Abalone" data set, but could not make the performance of IVNs competitive with Gaussian processes (which reached an MSE of about 0.29, compared to 0.39 for IVNs) and thus discarded that data set as a good way to compare IVNs to other neural models.

which specify Gaussian distributions $p(y_i|x_i) = \mathcal{N}(\mu_i, \sigma_i^2)$. To generate a data set we sample $\{x_i\}$ points uniformly from the input range $[-0.1, 1]$, and then sample $\{y_i\}$ accordingly. To compare plain neural networks (NN), density networks (DN), networks trained with fast dropout (FD) and implicit variance networks (IVN), we constructed a setting which is far from tailored towards neural networks: very little data.

The data set contained only 50 points for training, 10 for validation and 50 additional points to asses the performance as a test set. We trained the networks for 5000 epochs with batch size either $10, 25, 50$ and $10, 25, 50$ or 100 hidden units.

Discussion Notably, all methods except ours perform as bad as expected for a neural network model in this setting. While the density networks are en par with our method in terms of mean squared error, they overfit extremly with respect to their predictive distribution. Neural networks neither trained classically nor with fast dropout achieve good results; the variance of fast dropout seems to be meaningless, which is not surprising as it is not trained.

Table 1. Results on the toy benchmark and the sarcos data set

	Toy		**Sarcos**	
Method	MSE	NLL	MSE	NLL
NN	4.2395	2.2694	0.0047	-1.1893
DN	3.8706	9.7303	0.0096	-1.2532
FD	4.3491	43486.7	0.0065	1.2667
IVN	3.8985	1.6187	0.0079	-1.3606

Table 2. Hyper parameter ranges common over different data sets

Hyper parameter	Choices
#hidden layers	$1, 2, 3$
Transfer function	rectifier, sigmoid
Step rate	$10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}$
Momentum	$0, 0.5, 0.9, 0.99, 0.999$
Decay	$0.7, 0.8, 0.9$
Input variance	$0, 0.1, 0.2$
Variance offset B	$0, 0.5, 1$

3.2 Sarcos: Inverse Robot Dynamics

We evaluated the models under consideration on a standard benchmark for learning robot inverse dynamics, the "Sarcos" data set. We trained the networks for 500 epochs and picked the batch size from $\{64, 128, 256, 512\}$ and the number of hidden units from $\{50, 100, 200, 300\}$. We want to stress several observations. For one, IVNs seem to be the best choice if one is interested in good performance of both MSE and NLL. Secondly, plain neural networks perform surprisingly well in our experiments. While both Gaussian processes and LWPR models have different advantages compared to neural networks (model uncertainty and efficient incremental online learning, respectively) our experiments show that both are outperformed in terms of predictive quality.

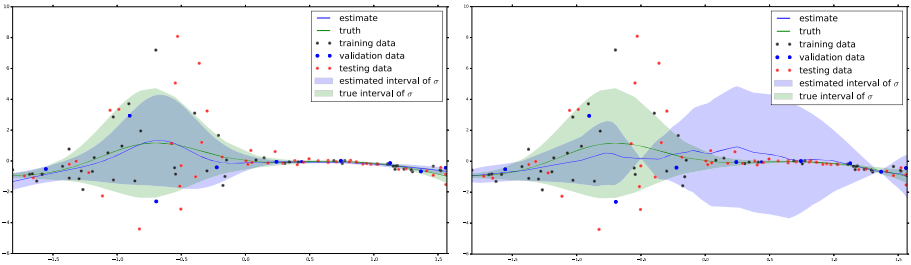


Fig. 1. Predictive distributions of IVNs and DNs on the toy benchmark

4 Conclusion and Acknowledgements

We presented a novel method to estimate predictive distributions via deep neural networks that plays nicely with fast dropout. The results are competitive or superior to other neural approaches in our experiments and en par with Gaussian processes and LWPR in a robotics task.

S. Urban was supported by German Research Foundation (DFG) SPP 1527 Autonomes Lernen.

References

1. Wang, S., Manning, C.: Fast dropout training. In: Proceedings of the 30th International Conference on Machine Learning (ICML 2013), pp. 118–126 (2013)
2. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 25, 1106–1114 (2012)
3. Dahl, G.E., Yu, D., Deng, L., Acero, A.: Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Transactions on Audio, Speech, and Language Processing 20(1), 30–42 (2012)
4. Larochelle, H., Erhan, D., Courville, A., Bergstra, J., Bengio, Y.: An empirical evaluation of deep architectures on problems with many factors of variation. In: Proceedings of the 24th International Conference on Machine Learning, pp. 473–480. ACM (2007)
5. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
6. Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3642–3649. IEEE (2012)
7. Zeiler, M., Ranzato, M., Monga, R., Mao, M., Yang, K., Le, Q., Nguyen, P., Senior, A., Vanhoucke, V., Dean, J., et al.: On rectified linear units for speech processing, ICASSP (2013)
8. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
9. Neal, R.M.: Connectionist learning of belief networks. Artificial Intelligence 56(1), 71–113 (1992)

10. Bengio, Y., Thibodeau-Laufer, É.: Deep generative stochastic networks trainable by backprop (2013)
11. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature 323(6088), 533–536 (1986)
12. Tang, Y., Salakhutdinov, R.: A new learning algorithm for stochastic feedforward neural nets (2013)
13. Bengio, Y.: Estimating or propagating gradients through stochastic neurons. arXiv preprint arXiv:1305.2982 (2013)
14. Salakhutdinov, R., Hinton, G.: Using deep belief nets to learn covariance kernels for gaussian processes. Advances in Neural Information Processing Systems 20, 1249–1256 (2008)
15. Uria, B., Murray, I., Renals, S., Richmond, K.: Deep architectures for articulatory inversion. In: Proceedings of Interspeech (2012)
16. Bishop, C.M.: Mixture density networks (1994)
17. Werbos, P.: Beyond regression: New tools for prediction and analysis in the behavioral sciences (1974)
18. Le Cun, Y.: Learning process in an asymmetric threshold network. In: Disordered Systems and Biological Organization, pp. 233–240. Springer (1986)
19. Bishop, C.M., et al.: Pattern recognition and machine learning, vol. 1. Springer, New York (2006)
20. Julier, S.J., Uhlmann, J.K.: New extension of the kalman filter to nonlinear systems. In: AeroSense 1997 International Society for Optics and Photonics, pp. 182–193 (1997)
21. Vijayakumar, S., D'souza, A., Schaal, S.: Incremental online learning in high dimensions. Neural Computation 17(12), 2602–2634 (2005)
22. Rasmussen, C.E.: Gaussian processes for machine learning. Citeseer (2006)
23. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.-R.: Efficient backProp. In: Orr, G.B., Müller, K.-R. (eds.) NIPS-WS 1996. LNCS, vol. 1524, pp. 9–50. Springer, Heidelberg (1998)
24. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. The Journal of Machine Learning Research 13, 281–305 (2012)
25. Tieleman, T., Hinton, G.: Lecture 6.5 - rmsprop: Divide the gradient by a running average of its recent magnitude. In: COURSERA: Neural Networks for Machine Learning (2012)
26. Sutskever, I.: Training Recurrent Neural Networks. PhD thesis, University of Toronto (2013)
27. Sutskever, I., Martens, J., Dahl, G., Hinton, G.: On the importance of initialization and momentum in deep learning (2013)
28. Le, Q.V., Smola, A.J., Canu, S.: Heteroscedastic gaussian process regression. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 489–496. ACM (2005)

One-Side Probability Machine: Learning Imbalanced Classifiers Locally and Globally

Rui Zhang and Kaizhu Huang

Xi'an Jiaotong-Liverpool University,
SIP, Suzhou, 215123, China
{Rui.Zhang02,Kaizhu.Huang}@xjtlu.edu.cn

Abstract. Imbalanced learning is a challenged task in machine learning, where the data associated with one class are far fewer than those associated with the other class. In this paper, we propose a novel model called One-Side Probability Machine (OSPM) able to learn from imbalanced data rigorously and accurately. In particular, OSPM can lead to a rigorous treatment on biased or imbalanced classification tasks, which is significantly different from previous approaches. Importantly, the proposed OSPM exploits the reliable global information from one side only, i.e., the majority class , while engaging the robust local learning [2] from the other side, i.e., the minority class. Such setting proves much effective than other models such as Biased Minimax Probability Machine (BMPM). To our best knowledge, OSPM presents the first model capable of learning from imbalanced data both locally and globally. Our proposed model has also established close connections with various famous models such as BMPM and Support Vector Machine. One appealing feature is that the optimization problem involved can be cast as a convex second order conic programming problem with a global optimum guaranteed. A series of experiments on three data sets demonstrate the advantages of our proposed method against four competitive approaches.

1 Introduction

Learning classifiers from imbalanced data is a challenged topic, arising very often in practice. Within this context, almost all the instances are labelled as one class, while far fewer instances are labelled as the other class, usually the more important class. Traditional classifiers seeking accurate performance over all the instances are not suitable to deal with imbalanced learning tasks. Obviously, such methods tend to classify all the data into the majority class, which is usually the less important class and hence would lead to serious problems in imbalanced data classification.

In the literature, there are various methods to deal with imbalanced data [8]. Among them are the methods of sampling, the methods of moving the decision thresholds, and the methods of adjusting the cost matrix. The first type of methods aims to reduce the data imbalance by "down-sampling" (removing) instances from the majority class or "up-sampling" (duplicating) the training

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 140–147, 2013.

instances from the minority class or both. The second type of methods tries to tune the decision threshold to impose a bias on the minority class. Similarly, the third type of methods improves the prediction performance by adapting the weight (cost) for each class.

It is argued that none of the methods is rigorous or systematic when handling imbalanced data [9,5,1]. Specifically, for the sampling method, either up- or down-sampling is unsuitable: up-sampling will introduce noise, while down-sampling the data will lose information. Moreover, it is usually difficult to know what a proportion should be sampled in order to incorporate a good bias. For these reasons, Provost states it as an open problem whether simply varying the skewness of the data distribution can improve prediction performance systematically [9]. For the method of adjusting the cost matrix or adapting weights, similar problems are also encountered, i.e., they are hard to establish direct connections between the cost matrix or the weights and the biased classification quantitatively. To impose a suitable bias towards the important class, they have to adapt these factors by trials. Therefore, these methods cannot rigorously handle imbalanced data.

In order to rigorously handle imbalanced data, a Biased Minimax Probability Machine (BMPM) [1] was proposed. In comparison with previous methods, BMPM takes advantages of the worst-case probability theory and establishes an explicit connection between the classification accuracy and the bias. It thus offers an elegant way to incorporate the bias into classification by directly controlling the real accuracy rather than intermediate factors. However, BMPM relies on exclusively certain global information for each class, i.e., the first and second order moments of data. Within the context of imbalanced learning, these global information, in particular, the second order moment for the minority data will be extremely unreliable due to its limited number.

Different from all the above mentioned approaches, in this paper, we propose a new model called One-Side Probability Machine (OSPM) which enjoys both the merits of BMPM and one important local learning method, i.e., Support Vector Machine (SVM).[1] When compared with the above-mentioned traditional three types of methods, OSPM neither duplicates or removes samples from data nor tunes those intermediate factors. It is also different from BMPM in that the proposed OSPM exploits the reliable global information from one side only, i.e., the majority class, while engaging the robust local learning [3] from the other side, i.e., the minority class. Similar to BMPM, OSPM presents a rigorous approach for imbalanced learning by directly tuning the accuracy of the majority class, while solving elegantly the limitation of BMPM by combining the thoughts of local learning for the minority class. This approach is appealing both theoretically and empirically on many aspects: (1) to our best knowledge, it is the first model capable of learning from imbalanced data both locally and globally; (2) it enjoys close theoretical connections with many other famous models, e.g., Maxi-Min Margin Machine [3], BMPM, and SVM; (3) an explicit connection

[1] Readers could refer to [3] on why SVM is regarded as one typical local learning model.

between the classification accuracy and the bias is still immediately available, making the model both rigorous and physically meaningful; (4) the involved optimization problem can be cast as a convex second order conic programming problem, which can be efficiently solved with a global optimum guaranteed.

2 Notation and Biased Minimax Probability Machine

Notation. The notation of this paper largely follows that used in [1,3]. For simplicity, we only consider the binary classification problem in this paper. Suppose two random d-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$ represent two classes of data, where $\mathbf{x}$ belongs to the family of distributions with a given mean $\bar{\mathbf{x}}$ and a covariance $\Sigma_{\mathbf{x}}$, denoted as $\mathbf{x} \sim (\bar{\mathbf{x}}, \Sigma_{\mathbf{x}})$; similarly, $\mathbf{y}$ belongs to the family of distributions with a given mean $\bar{\mathbf{y}}$ and a covariance $\Sigma_{\mathbf{y}}$, denoted as $\mathbf{y} \sim (\bar{\mathbf{y}}, \Sigma_{\mathbf{y}})$. Here $\mathbf{x}$, $\mathbf{y}$, $\bar{\mathbf{x}}$, $\bar{\mathbf{y}} \in \mathbb{R}^d$, and $\Sigma_{\mathbf{x}}$, $\Sigma_{\mathbf{y}} \in \mathbb{R}^{d\times d}$. In this paper, the class $\mathbf{x}$ also represents the important or minority class and the class $\mathbf{y}$ represents the corresponding less important or majority class. Given a training data set $\mathbb{D}$ consisting of $n_{\mathbf{x}}$ and $n_{\mathbf{y}}$ samples for class $\mathbf{x}$ and $\mathbf{y}$ respectively, i.e., $\mathbb{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_{\mathbf{x}}}, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_{\mathbf{y}}}\}$, the objective of classification can be informally described to find an hyperplane $\{\mathbf{w}, b\}$ ($\mathbf{w} \neq \mathbf{0}$) so that a future data sample $\mathbf{z}$ could be correctly classified. Namely, if $\mathbf{w}^\top \mathbf{z} + b \geq 0$, $\mathbf{z}$ is classified as class $\mathbf{x}$, otherwise as class $\mathbf{y}$. Within the context of imbalanced classification where the class $\mathbf{x}$ is more important than the class $\mathbf{y}$ (usually $n_{\mathbf{x}} \ll n_{\mathbf{y}}$), we should maximize the accuracy of $\mathbf{x}$ while maintaining acceptable the accuracy of $\mathbf{y}$.

Biased Minimax Probability Machine. BMPM [1] involves the following optimization problem:

$$\max_{\alpha,\beta,b,\mathbf{w}\neq\mathbf{0}} \quad \alpha \quad \text{s.t.} \quad \inf_{\mathbf{x}\sim(\bar{\mathbf{x}},\Sigma_{\mathbf{x}})} \Pr\{\mathbf{w}^T\mathbf{x} + b \geq 0\} \geq \alpha \,, \tag{1}$$

$$\inf_{\mathbf{y}\sim(\bar{\mathbf{y}},\Sigma_{\mathbf{y}})} \Pr\{\mathbf{w}^T\mathbf{y} + b \leq 0\} \geq \beta \,, \qquad \beta \geq \beta_0 \,. \tag{2}$$

α means the lower bound of the probability (accuracy) for the classification of future cases of the class $\mathbf{x}$ with respect all distributions with the mean and covariance as $(\mathbf{x}, \Sigma_{\mathbf{x}})$; in other words, α is the worst-case accuracy for the class $\mathbf{x}$. Similarly, β is the lower bound of the accuracy of the class $\mathbf{y}$. This optimization achieves to maximize the accuracy (probability α) for the biased class $\mathbf{x}$ while simultaneously maintaining the class $\mathbf{y}$'s accuracy at an acceptable level β_0.

BMPM obtains the biased classifier by directly controlling the real accuracy β_0 over the majority class. This presents a rigorous approach which distinguishes it from traditional methods adapting intermediate parameters (cost matrices or weights) based on trials. However, the optimization of BMPM exclusively relies on the global information, i.e., the means $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ and the covariances $\Sigma_{\mathbf{x}}$, $\Sigma_{\mathbf{y}}$. In imbalanced learning, the more important class, or the minority class $\mathbf{x}$ contains very limited number of samples, especially when compared with that of the majority class $\mathbf{y}$. This leads the mean $\bar{\mathbf{x}}$ and the covariance $\Sigma_{\mathbf{x}}$ are extremely unreliable. Such drawback may limit the performance of BMPM. Targeting this problem, we then propose the novel One-Side Probability Machine.

3 One-Side Probability Machine

Model Definition. The model of OSPM is formulated as

$$\min_{\mathbf{w},b,\xi} \ \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n_{\mathbf{x}}} \xi_i \qquad \text{s.t.} \tag{3}$$

$$\mathbf{w}^\top \mathbf{x}_i + b \geq 1 - \xi_i, \quad \forall i, 1 \leq i \leq n_{\mathbf{x}} \ , \tag{4}$$

$$\inf_{\mathbf{y}\sim(\bar{\mathbf{y}},\Sigma_{\mathbf{y}})} \Pr\{\mathbf{w}^\top \mathbf{y} + b \leq 0\} \geq \beta_0 \ . \tag{5}$$

In the above, we are trying to maintain the real test accuracy of the majority class $\mathbf{y}$ using the constraint (5). Specifically, given the reliable mean $\bar{\mathbf{y}}$, and covariance $\Sigma_{\mathbf{y}}$, the left part of (5) represents the worst-case accuracy for class $\mathbf{y}$ associated with the hyperplane $\{\mathbf{w}, b\}$ among all possible distributions with the given mean and the covariance for class $\mathbf{y}$. This worst-case accuracy is required to be greater than a pre-defined accuracy β_0. For example, β_0 could be set to 60%, meaning that we would like to maintain the accuracy of the majority class acceptable at the level of 60%. In comparison, the objective function and the constraint of (4) simply borrows the idea of SVM: the minority class $\mathbf{x}$ should be classified as correctly as possible by satisfying $\mathbf{x}$ in (4) and minimizing the "error" $\sum_{i=1}^{n_{\mathbf{x}}} \xi_i$ plus one regularization term $\frac{1}{2}||\mathbf{w}||^2$ in (3). Note that the majority data affect the decision plane by the single constraint (5). This significantly moderates the negative influence caused by its overwhelming number of samples against the minority class.

Observed from the above model, OSPM exploits a one-side probabilistic setting only, which motivates the name of our model. Namely, we require its worst-case accuracy is maintained at the accuracy of β_0 merely for the majority or less important side (class), since the global information $\bar{\mathbf{y}}$, and covariance $\Sigma_{\mathbf{y}}$ can be reliably estimated. As the number of the more important side (class) is very limited, the global information associated with $\mathbf{x}$ will not be reliable. We hence engage the robust local information, i.e., each single data point in (4). The setting of focusing on the local information is called local learning [3,2]. In brief, the proposed novel model utilizes both reliable global information and robust local information. To the best of our knowledge, this is the first model which learns from imbalanced data both locally and globally.

Connection with Biased Minimax Probability Machine. It is easily observed that the proposed One-Side Probability Machine is closely linked with BMPM. BMPM focuses on using global probabilistic information from both sides, i.e., the means and covariances from both classes, while our proposed model merely exploits one side probabilistic information from the majority class, which is usually reliable. For the minority class, OSPM takes advantages of the local learning, focusing on every single local point instead of unreliable global information. This presents one big advantage over BMPM.

Connection with Support Vector Machine. The model of SVM can be described as

$$\min_{\mathbf{w}\neq\mathbf{0},b,\xi} \ \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n_\mathbf{x}+n_\mathbf{y}} \xi_i \qquad \text{s.t.} \tag{6}$$

$$\mathbf{w}^\top \mathbf{x}_i + b \geq 1 - \xi_i, \quad \forall i, 1 \leq i \leq n_\mathbf{x} \ , \tag{7}$$

$$-\mathbf{w}^\top \mathbf{y}_i - b \geq 1 - \xi_{j+n_\mathbf{x}}, \quad \forall j, 1 \leq j \leq n_\mathbf{y} \ . \tag{8}$$

When the number of minority class is limited, the above optimization maximizing the full range of instances tends to classify all the samples as the majority class. To alleviate the problem, the item of $C\sum_{i=1}^{n_\mathbf{x}+n_\mathbf{y}} \xi_i$ is usually replaced as $C_\mathbf{x}\sum_{i=1}^{n_\mathbf{x}} \xi_i + C_\mathbf{y}\sum_{i=1}^{n_\mathbf{y}} \xi_{i+n_\mathbf{x}}$ so that different weights could be given to different classes. However, without explicit connection between the trade-off parameters $C_\mathbf{x}$, $C_\mathbf{y}$ and the accuracy, it is difficult to precisely impose bias to the important class. In comparison, the proposed OSPM directly controls the real accuracy given by the parameter β_0, presenting a more rigorous treatment on imbalanced data.

Connection with Maxi-Min Margin Machine. Maxi-Min Margin Machine (M^4) presents the first model capable of learning from *balanced* data locally and globally. Considering both the global information obtained from covariance matrices $\Sigma_\mathbf{x}$ and $\Sigma_\mathbf{y}$ as well as local information from each local point, M^4 successfully unifies the SVM, Minimax Probability Machine, and Linear Discriminant Analysis into a common framework. However, M^4 merely targets the balanced data set. Similarly, given highly imbalanced data, the mean and the covariance of the minority data could still be unreliably estimated. In comparison, the proposed OSPM is designed specially for imbalanced data. It utilizes reliable global information from the majority class, while engaging local information of each specific data point from the minority class. To our best knowledge, OSPM is the first model capable of learning from *imbalanced* data both locally and globally.

Optimization

Lemma 1. *Given* $\mathbf{w} \neq \mathbf{0}$ *and* b*, such that* $\mathbf{w}^T\mathbf{y} + b \leq 0$ *and* $\beta \in [0, 1)$*, the condition* $\inf_{\mathbf{y}\sim(\overline{\mathbf{y}},\Sigma_\mathbf{y})} \mathbf{Pr}\{\mathbf{w}^T\mathbf{y} + b \leq 0\} \geq \beta$ *holds if and only if* $-b - \mathbf{w}^T\overline{\mathbf{y}} \geq \kappa(\beta)\sqrt{\mathbf{w}^T\Sigma_\mathbf{y}\mathbf{w}}$ *with* $\kappa(\beta) = \sqrt{\frac{\beta}{1-\beta}}$.

The lemma can be proved according to the Marshall and Olkin Theory and the Lagrangian Multiplier theory. The details can be found in [6] and [2]. By using Lemma 1, we can transform the OSPM optimization problem as follows:

$$\min_{\mathbf{w},b,\xi} \ \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n_\mathbf{x}} \xi_i \qquad \text{s.t.} \tag{9}$$

$$\mathbf{w}^\top \mathbf{x}_i + b \geq 1 - \xi_i, \quad \forall i, 1 \leq i \leq n_\mathbf{x} \tag{10}$$

$$-b - \mathbf{w}^T\overline{\mathbf{y}} \geq \kappa(\beta_0)\sqrt{\mathbf{w}^T\Sigma_\mathbf{y}\mathbf{w}} \tag{11}$$

where $\kappa(\beta_0) = \sqrt{\frac{\beta_0}{1-\beta_0}}$. (11) is directly obtained from (5) by using Lemma 1.

The above optimization problem can easily be verified to be a jointly convex problem with respect to $\mathbf{w}$, b, and ξ or in particular, one Second Order Conic Programming Problem (SOCP) [7]. First, the objective function (9) is quadratically convex. Moreover, constraint (10) is linear. To better examine the convexity of (11), we could always pre-process the data associated with $\mathbf{y}$ so that the mean $\bar{\mathbf{y}}$ can be shifted to zero. It will then be easily seen that (11) is a typical second-order convex constraint. In summary, the involved optimization problem is a convex second-order conic programming problem, which could be solved efficiently by many off-the-shelf softwares, e.g., Sedumi or CVX.

4 Experiments

In this section, we evaluate the performance of our proposed OSPM on three real world imbalanced data sets in comparison with four other competitive approaches, i.e., the Naive Bayesian (NB) classifier, the k-Nearest Neighbor (k-NN) method, the decision tree classifier C4.5, and the BMPM model in both the linear (BMPML) and Gaussian (BMPMG) kernels. For brevity, we name the linear and the Gaussian kernel versions of our proposed model as OSPML and OSPMG. Following previous work [1,4], we engage the criterion of the Maximum Sum (MS) of the accuracies on the majority class (True Negative) and the minority class (True Positive), and the area of Receiver Operating Characteristic (ROC) curve to conduct comparisons.

Evaluations on the Recidivism Dataset. A training set where 570 (27.5%) individuals were recidivists and 970 (72.5%) were not and a test set with $1,151$ individuals as recidivists were released for the recidivism data set in the North Carolina prison system [1]. We compare the performance of our proposed OSPM model in both the linear (OSPML) and the Gaussian kernel setting (OSPMG), with the above-mentioned four approaches. We run k-NN methods for $k = 1, 3, 5, \ldots, 21$, but we only present the best three results for brevity. We present the experimental results based on the MS and the ROC area criteria in Table 1 where Tn and Tp represent true negative and true positive respectively. The average, i.e., $\frac{Tp+Tn}{2}$ represents the MS criterion. To be more comparable, we show the average of the accuracy for each class when each classifier attains the point

Table 1. Performance on Recidivism using the MS and ROC criteria

Method	Tn (%)	Tp (%)	MS (%)	Method	Area Under ROC Curve (%)
NB	61.8	63.8	62.7	NB	66.5
k-NN(9)	62.6	54.6	58.6	k-NN(11)	61.6
k-NN(11)	62.4	55.4	58.9	k-NN(13)	61.9
k-NN(13)	55.7	62.0	58.9	k-NN(17)	61.5
C4.5	74.1	49.0	61.5	C4.5	63.8
BMPML	70.4	57.5	63.9	BMPML	68.4
BMPMG	72.0	57.8	64.9	BMPMG	68.0
OSPML	70.9	57.5	**64.2**	OSPML	**68.8**
OSPMG	72.9	57.5	**65.2**	OSPMG	**68.9**

Table 2. Comparison based on the MS criterion on Breast-cancer and Heart

breast-cancer (%)				heart (%)			
Method	Specificity	Sensitivity	MS	Method	Specificity	Sensitivity	MS
k-NN(11)	99.0 ± 0.5	96.2 ± 0.3	97.6 ± 0.3	k-NN(17)	76.5 ± 0.3	88.4 ± 0.2	82.5 ± 0.3
k-NN(17)	98.6 ± 0.9	96.6 ± 0.6	97.6 ± 0.5	k-NN(7)	77.5 ± 0.4	88.4 ± 0.4	83.0 ± 0.4
k-NN(7)	97.2 ± 0.7	97.5 ± 0.5	97.4 ± 0.6	k-NN(15)	75.1 ± 0.3	86.5 ± 0.4	80.8 ± 0.4
NB	93.7 ± 0.6	97.2 ± 0.5	95.4 ± 0.5	NB	78.6 ± 0.5	80.2 ± 0.3	79.4 ± 0.4
C4.5	93.8 ± 0.7	95.8 ± 0.7	94.8 ± 0.7	C4.5	88.3 ± 0.2	70.7 ± 0.2	79.5 ± 0.2
BMPML	96.8 ± 0.3	98.7 ± 0.2	97.8 ± 0.2	BMPML	85.5 ± 0.4	81.6 ± 0.1	83.5 ± 0.4
BMPMG	96.1 ± 0.2	99.2 ± 0.0	97.6 ± 0.2	BMPMG	84.0 ± 0.1	85.7 ± 0.2	84.9 ± 0.3
OSPML	96.9 ± 0.3	99.2 ± 0.2	**98.1 ± 0.2**	OSPML	85.8 ± 0.3	82.2 ± 0.2	**84.0 ± 0.2**
OSPMG	96.3 ± 0.2	99.6 ± 0.1	**98.0 ± 0.2**	OSPMG	84.7 ± 0.2	86.1 ± 0.3	**85.4 ± 0.3**

Table 3. Comparison based on the ROC analysis

breast-cancer (%)		heart (%)	
Method	Area under ROC Curve	Method	Area under ROC Curve
k-NN(11)	99.1 ± 0.6	k-NN(17)	87.0 ± 0.4
k-NN(17)	99.0 ± 1.0	k-NN(7)	86.9± 0.5
k-NN(7)	98.9 ± 0.8	k-NN(15)	86.0 ± 0.4
NB	98.4 ± 0.6	NB	81.6 ± 0.3
C4.5	97.6 ± 1.2	C4.5	83.0± 0.4
BMPML	99.5 ± 0.2	BMPML	88.1± 0.6
BMPMG	99.6 ± 0.2	BMPMG	89.3± 0.4
OSPML	**99.6 ± 0.2**	OSPML	**88.6± 0.5**
OSPMG	**99.7 ± 0.3**	BMPMG	**89.8± 0.4**

of the maximum sum. As observed, in terms of both MS and the area of ROC, the OSPML and OSPMG achieve the best performance. In comparison, except OSPM, the BMPML and BMPMG achieve the highest results among all the remaining algorithms. Our method further boosts the performance of BMPML and BMPMG by solving the limitation of BMPM.

Evaluations on Disease Data Sets. Diagnosing diseases contains a very similar characteristic to the imbalanced learning, since one class, often the disease class needs to be given more bias than the other class. Therefore, the above discussed model modifications will be automatically applicable for this kind of tasks. In the following, we evaluate the performance of OSPM on two disease data sets, namely, the Breast-cancer data and the Heart-disease data set, obtained from UCI machine learning repository. In the context of diagnosing diseases, the true positive rate is usually called sensitivity, while the true negative rate is called specificity. Therefore, we should maximize the sensitivity while maintaining the specificity acceptable. We randomly split the data for each data set into a training set with 80% data and a test set with 20% data. We then construct classifiers based on the training data set and perform evaluations on the test data set. We repeat this procedure ten times and use the average of the results as the performance metric.

We present the results based on the MS criterion in Table 2 for the breast-cancer and the heart disease data set. As observed, the proposed model also demonstrates a superiority to other four models. We also show the experimental

results based on the ROC area analysis in Table 3. Similarly, our proposed OSPML and OSPMG once again outperform the other comparison algorithms.

5 Conclusion

In this paper, we have proposed a novel model called One-Side Probability Machine (OSPM) that is specially designed for imbalanced learning. Different from the previous approaches, OSPM can lead to a rigorous treatment on biased classification tasks. Importantly, the proposed OSPM exploits the reliable global information from one side only, i.e., the majority class , while engaging the robust local learning [2] from the other side, i.e., the minority class. A series of experiments demonstrated the effectiveness of our new model.

References

1. Huang, K., Yang, H., King, I., Lyu, M.R.: Learning classifiers from imbalanced data based on biased minimax probability machine. In: Proceedings of CVPR, vol. 2, pp. 558–563 (2004)
2. Huang, K., Yang, H., King, I., Lyu, M.R.: Machine Learning: Modeling Data Locally and Gloablly. Springer (2008) ISBN 3-5407-9451-4
3. Huang, K., Yang, H., King, I., Lyu, M.R.: Maxi-min margin machine: Learning large margin classifiers globally and locally. IEEE Transactions on Neural Networks 19, 260–272 (2008)
4. Huang, K., Yang, H., King, I., Lyu, M.R.: Imbalanced learning with biased minimax probability machine. IEEE Transactions on systems, Man and Cybernetics, Part B 36(4), 913–923 (2006)
5. Huang, K., Yang, H., King, I., Lyu, M.R., Chan, L.: The minimum error minimax probability machine. Journal of Machine Learning Research 5, 1253–1286 (2004)
6. Lanckriet, G.R.G., Ghaoui, L.E., Bhattacharyya, C., Jordan, M.I.: A robust minimax approach to classification. Journal of Machine Learning Research 3, 555–582 (2002)
7. Lobo, M., Vandenberghe, L., Boyd, S., Lebret, H.: Applications of second order cone programming. Linear Algebra and its Applications 284, 193–228 (1998)
8. Maloof, M.A., Langley, P., Binford, T.O., Nevatia, R., Sage, S.: Improved rooftop detection in aerial images with machine learning. Machine Learning 53, 157–191 (2003)
9. Provost, F.: Learning from imbanlanced data sets. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence, AAAI 2000 (2000)

Structure Preserving Low-Rank Representation for Semi-supervised Face Recognition

Yong Peng[1], Suhang Wang[2], Shen Wang[2], and Bao-Liang Lu[1,3,⋆]

[1] Center for Brain-Like Computing and Machine Intelligence,
Department of Computer Science and Engineering,
Shanghai Jiao Tong Unviersity, Shanghai 200240 China
[2] Department of Electrical Engineering and Computer Science,
University of Michigan, Ann Arbor, 48109 USA
[3] MOE-Microsoft Key Lab. for Intelligent Computing and Intelligent Systems
Shanghai Jiao Tong Unviersity, Shanghai 200240 China
bllu@sjtu.edu.cn

Abstract. Constructing an informative and discriminative graph plays an important role in the graph based semi-supervised learning methods. Among these graph construction methods, low-rank representation based graph, which calculates the edge weights of both labeled and unlabeled samples as the low-rank representation (LRR) coefficients, has shown excellent performance in semi-supervised learning. In this paper, we additionally impose twofold constraints (*local affinity* and *distant repulsion*) on the LRR graph. The improved model, termed structure preserving LRR (SPLRR), can preserve the local geometrical structure but without distorting the distant repulsion property. Experiments are taken on three widely used face data sets to investigate the performance of SPLRR and the results show that it is superior to some state-of-the-art semi-supervised graphs.

Keywords: Structure preserving, Low-rank representation, Semi-supervised learning, Face recognition.

1 Introduction

Recently, semi-supervised learning (SSL) has received increasing attention because it can utilize both limited labeled samples and rich yet unlabeled samples. The currently available semi-supervised methods can be roughly categorized into four groups: generative models, low-density separation models, heuristic models and graph-based models. In this paper, we focus our work on graph-based SSL due to its empirical success in practice and computational efficiency.

Graph-based SSL relies on using a graph $G = (V, E, W)$ to represent data structure, where V is a set of vertices in which each vertex represents a data point, $E \subseteq V \times V$ is a set of edges connecting related vertices and W is an adjacency matrix recording the pairwise weights between vertices. Usually, the graph

⋆ Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 148–155, 2013.

is constructed using relationship of domain knowledge or similarity of samples. Upon the graph is constructed, each sample spreads its label information to its neighbors until a global stable state is achieved on the whole data set. Thus, both labeled and unlabeled samples remarkably affect the construction of graphs and how to construct a good graph for representing the data structure is critical for graph-based SSL. Recently, some commonly used graphs have been well investigated, such as k nearest neighbors graph [10], graph for label propagation based on linear neighborhoods (LNP) [11], ℓ^1 graph [5], sparse probability graph (SPG) [6] and so on.

The ℓ^1 graph is motivated by which each datum can be reconstructed by the sparse linear superposition of the training data [5] and the sparse reconstruction coefficients are derived by solving an ℓ^1 optimization problem. Differing from the sparse representation which enforces the representation coefficients to be sparse, the semi-supervised low-rank representation graph (LRR) was proposed for pattern classification [12]. However, the low rankness constraint can only capture the global mixture of subspaces structure while ignoring the local structure of data. To compensate the drawback of LRR graph, we propose a structure preserving low-rank representation based graph, which is imposed on twofold constraints: local affinity and distant repulsion. Therefore, the proposed structure preserving low-rank representation can properly preserve the local affinity structure without distorting the distant repulsion property.

The remainder of this paper is organized as follows. We present a brief review of low-rank representation in section 2. In section 3, we propose the formulation of structure preserving low-rank representation (SPLRR) model and its implementation which is based on the inexact ALM algorithm. Section 4 shows the semi-supervised classification method used in this paper. Experiments on three widely used face databases for evaluating the performance of SPLRR are illustrated in section 5. Conclusion is given in section 6.

2 Low-Rank Representation

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ be a set of data points in d-dimensional space. We try to represent each sample in $\mathbf{X}$ based on the dictionary $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_m] \in \mathbb{R}^{d \times m}$ using $\mathbf{X} = \mathbf{AZ}$, where $\mathbf{Z}$ is the representation coefficients matrix. When the dictionary $\mathbf{A}$ is over-complete, there will be many solutions to this problem. LRR seeks a lowest-rank solution by solving the following problem [8] (we use the data matrix $\mathbf{X}$ itself as dictionary):

$$\min_{\mathbf{z}} \quad \text{rank}(\mathbf{Z}), \ \ s.t. \ \mathbf{X} = \mathbf{XZ}. \tag{1}$$

The optimal solution to (1) is called the "lowest-rank representations" of data $\mathbf{X}$ w.r.t. a dictionary $\mathbf{X}$. However, this problem is NP-hard to solve due to the discrete nature of *rank* function. Fortunately, we can convert (1) to following convex optimization problem instead based on the work of [3]:

$$\min_{\mathbf{Z}} \quad ||\mathbf{Z}||_\star, \quad s.t. \ \mathbf{X} = \mathbf{XZ}, \tag{2}$$

where $||\cdot||_\star$ denotes the nuclear norm of a matrix [2], i.e., the sum of the singular values of the matrix.

In real-world applications, data points are often noisy or even grossly corrupted. Therefore, the corrupted data can be separated to two parts, i.e., $\mathbf{X} = \mathbf{XZ} + \mathbf{E}$. Thus, the affinity matrix $\mathbf{Z}$ can be obtained by solving the following problem:

$$\min_{\mathbf{Z},\mathbf{E}} \quad ||\mathbf{Z}||_\star + \lambda||\mathbf{E}||_\ell, \quad s.t.\ \mathbf{X} = \mathbf{XZ} + \mathbf{E}, \tag{3}$$

where $||\cdot||_\ell$ can be the $\ell_{2,1}$ or ℓ_1 norm (in this paper we choose ℓ_1 norm). The optimal solution $\mathbf{Z}^\star$ to problem (3) can be obtained via the Inexact Augmented Lagrange Multiplier Method (ALM) [7].

3 Structure Preserving Low-Rank Representation

In this section, we propose the structure preserving low-rank representation (SPLRR) model as well as its solution based on Inexact Augmented Lagrange Multiplier Method [7]. SPLRR can properly preserve the local affinity structure without distorting the distant repulsion property. The local affinity indicates the local neighborhood correlation, which means that if $\mathbf{x}_i$ and $\mathbf{x}_j$ are close in the original data space, their corresponding representation coefficients $\mathbf{z}_i$ and $\mathbf{z}_j$ should be also close in the transformed space. The distant repulsion property is inspired by the elastic embedding [4], which enforces the corresponding representation coefficients of distant data points in the original space to be kept distant in the transformed space.

3.1 Formulation of SPLRR

We first introduce the two constraints: *local affinity* and *distant repulsion*.

• Local affinity. To preserve the local geometrical structure in the coefficient space, one may naturally hope that, if two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ are close in the intrinsic manifold, their corresponding representation coefficients $\mathbf{z}_i$ and $\mathbf{z}_j$ should also be close to each other. This can be viewed as manifold assumption for smoothness transition. Several methods can achieve this manifold-like property and in this work we choose the graph regularization term which is similar to graph regularized non-negative matrix factorization (GNMF) in [1]:

$$\begin{aligned}\min_{\mathbf{Z}} \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}||\mathbf{z}_i - \mathbf{z}_j||^2 &= \sum_{i=1}^{n} d_{ii}\mathbf{z}_i^T\mathbf{z}_i - \sum_{i,j=1}^{n} w_{ij}\mathbf{z}_i^T\mathbf{z}_j \\ &= \mathrm{Tr}(\mathbf{ZDZ}^T) - \mathrm{Tr}(\mathbf{ZWZ}^T) = \mathrm{Tr}(\mathbf{ZL}_1\mathbf{Z}^T)\end{aligned} \tag{4}$$

where $\mathbf{D}$ is a diagonal matrix whose entries are column (or row, since $\mathbf{W}$ is symmetric) sums of $\mathbf{W}$, $d_{ii} = \sum_j w_{ij}$. We can compute the graph Laplacian $\mathbf{L}_1 = \mathbf{D} - \mathbf{W}$, w_{ij} is the measure of affinity between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the original space. Here, we use the 'HeatKernel' formulation:

$$w_{ij} = \begin{cases} \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2t^2), & \text{if } \mathbf{x}_i \in \mathcal{N}_{k1}(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathcal{N}_{k1}(\mathbf{x}_i), \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

where $\mathcal{N}_p(\mathbf{x}_i)$ denotes the set of p nearest neighbors of $\mathbf{x}_i$.

• Distant repulsion. This property enforces dissimilar data pairs in the original space to be far apart in the embedded space. Here we use the identical formulation with the 'local affinity' property as

$$\min_{\mathbf{Z}} \frac{1}{2} \sum_{i,j=1}^{n} s_{ij} ||\mathbf{z}_i - \mathbf{z}_j||^2 = \mathrm{Tr}(\mathbf{Z}\mathbf{L}_2\mathbf{Z}^T), \tag{6}$$

where $\mathbf{L}_2$ has similar property as $\mathbf{L}_1$ and

$$s_{ij} = \begin{cases} ||\mathbf{x}_i - \mathbf{x}_j||^2 \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2t^2), & \text{if } \mathbf{x}_i \in \mathcal{N}_{k2}(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathcal{N}_{k2}(\mathbf{x}_i), \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Integrating Eqs.(4), (6) into the low-rank representation model, we can get the SPLRR model as follows:

$$\min_{\mathbf{Z},\mathbf{E}} ||\mathbf{Z}||_\star + \lambda||\mathbf{E}||_1 + \alpha\mathrm{Tr}(\mathbf{Z}\mathbf{L}_1\mathbf{Z}^T) + \beta\mathrm{Tr}(\mathbf{Z}\mathbf{L}_2\mathbf{Z}^T) \;\; s.t. \;\; \mathbf{X} = \mathbf{X}\mathbf{Z} + \mathbf{E}. \tag{8}$$

3.2 Solution to SPLRR

Similar to [8], Eq.(8) can be transformed into the following equivalent problem by introducing the auxiliary variable $\mathbf{J}$:

$$\begin{aligned} \min_{\mathbf{Z},\mathbf{E}} &||\mathbf{J}||_\star + \lambda||\mathbf{E}||_1 + \alpha\mathrm{Tr}(\mathbf{Z}\mathbf{L}_1\mathbf{Z}^T) + \beta\mathrm{Tr}(\mathbf{Z}\mathbf{L}_2\mathbf{Z}^T) \\ &s.t. \;\; \mathbf{X} = \mathbf{X}\mathbf{Z} + \mathbf{E} \quad \text{and} \quad \mathbf{Z} = \mathbf{J}. \end{aligned} \tag{9}$$

In order to efficiently solve the optimization problem (9), the ALM method [7] is utilized. Thus, the Augmented Lagrange function w.r.t. (9) is:

$$\begin{aligned} \min_{\mathbf{Z},\mathbf{E},\mathbf{J},\mathbf{Y}_1,\mathbf{Y}_2} &||\mathbf{J}||_\star + \lambda||\mathbf{E}||_1 + \alpha\mathrm{Tr}(\mathbf{J}\mathbf{L}_1\mathbf{J}^T) + \beta\mathrm{Tr}(\mathbf{J}\mathbf{L}_2\mathbf{J}^T) + \langle\mathbf{Y}_1, \mathbf{X} - \mathbf{X}\mathbf{Z} - \mathbf{E}\rangle \\ &+\langle\mathbf{Y}_2, \mathbf{Z} - \mathbf{J}\rangle + \mu/2\left(||\mathbf{X} - \mathbf{X}\mathbf{Z} - \mathbf{E}||_F^2 + ||\mathbf{Z} - \mathbf{J}||_F^2\right). \end{aligned} \tag{10}$$

Obviously, we need to optimize this problem over one variable with others fixed. The subproblem w.r.t. each variable is convex and thus can provide corresponding unique optimal solutions. The optimization method to SPLRR is summarized in Algorithm 1. Note that, we give the relaxation of the objective when updating variable $\mathbf{J}$ and the derivation of Eq.(13) is as follows:

$$\begin{aligned} \mathcal{L} &= \min ||\mathbf{J}||_\star + \lambda||\mathbf{E}||_1 + \mathrm{Tr}\left(\mathbf{J}(\alpha\mathbf{L}_1 + \beta\mathbf{L}_2)\mathbf{J}^T\right) \\ &\leq \min ||\mathbf{J}||_\star + \lambda||\mathbf{E}||_1 + ||\mathbf{J}||_F \cdot ||\alpha\mathbf{L}_1 + \beta\mathbf{L}_2||_F \cdot ||\mathbf{J}||_F \\ &= \min ||\mathbf{J}||_\star + \lambda||\mathbf{E}||_1 + a\langle\mathbf{J}, \mathbf{J}\rangle \;\; (a \triangleq ||\alpha\mathbf{L}_1 + \beta\mathbf{L}_2||_F). \end{aligned} \tag{11}$$

For updating $\mathbf{J}$ while other variables fixed, we have

$$\begin{aligned} \mathbf{J} &= \arg\min ||\mathbf{J}||_\star + a\langle\mathbf{J}, \mathbf{J}\rangle - \langle\mathbf{Y}_2 + \mu\mathbf{Z}, \mathbf{J}\rangle + \mu/2\langle\mathbf{J}, \mathbf{J}\rangle \\ &= \arg\min ||\mathbf{J}||_\star + (a + \mu/2)\langle\mathbf{J}, \mathbf{J}\rangle - \mu\langle\mathbf{Z} + \mathbf{Y}_2/\mu, \mathbf{J}\rangle \\ &= \arg\min \frac{1}{2a+\mu}||\mathbf{J}||_\star + \frac{1}{2}||\mathbf{J} - \frac{\mu}{2a+\mu}\left(\mathbf{Z} + \frac{\mathbf{Y}_2}{\mu}\right)||_F^2. \end{aligned} \tag{12}$$

The SPLRR-based graph construction model is concluded in Algorithm 2.

Algorithm 1. Solving Problem (8) via Inexact ALM

Input: Data matrix $\mathbf{X}$; regularization parameters λ, α and β; parameters for constructing the affinity graph Laplacian and repulsion graph Laplacian.

Output: The affinity matrix $\mathbf{Z}$.

Initialization: set $\mathbf{Z} = \mathbf{J} = \mathbf{0}$, $\mathbf{E} = 0$, $\mathbf{Y}_1 = \mathbf{Y}_2 = \mathbf{0}$, $\mu = 10^{-6}$, $M = 10^{10}$, $\rho = 1.1$ and $\varepsilon = 10^{-8}$.

Construct the graph Laplacian $\mathbf{L}_1$ for local affinity and $\mathbf{L}_2$ for distant repulsion.

Repeat until converge:

- Updating $\mathbf{J}$:

$$\mathbf{J} = \arg\min \frac{1}{2a+\mu}||\mathbf{J}||_\star + \frac{1}{2}||\mathbf{J} - \frac{\mu}{2a+\mu}\left(\mathbf{Z} + \frac{\mathbf{Y}_2}{\mu}\right)||_F^2,\ a = ||\alpha\mathbf{L}_1 + \beta\mathbf{L}_2||_F \tag{13}$$

- Updating $\mathbf{Z}$: $\mathbf{Z} = (\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{X} - \mathbf{X}^T\mathbf{E} + \mathbf{J} + (\mathbf{X}^T\mathbf{Y}_1 - \mathbf{Y}_2)/\mu)$
- Updating $\mathbf{E}$: $\mathbf{E} = \arg\min \frac{\lambda}{\mu}||\mathbf{E}||_1 + \frac{1}{2}||\mathbf{E} - (\mathbf{X} - \mathbf{X}\mathbf{Z} + \mathbf{Y}_1/\mu)||_F^2$
- Updating multipliers: $\mathbf{Y}_1 = \mathbf{Y}_1 + \mu(\mathbf{X} - \mathbf{X}\mathbf{Z} - \mathbf{E}),\quad \mathbf{Y}_2 = \mathbf{Y}_2 + \mu(\mathbf{Z} - \mathbf{J})$
- Updating the parameter μ by $\mu = \min(\rho\mu, M)$
- Checking the convergence conditions

$$||\mathbf{X} - \mathbf{X}\mathbf{Z} - \mathbf{E}||_\infty < \varepsilon \ \text{ and } \ ||\mathbf{Z} - \mathbf{J}||_\infty < \varepsilon$$

Algorithm 2. Graph construction based on SPLRR

Input: Data marix $\mathbf{X}$, parameters for computing affinity graph Laplacian and repulsion graph Laplacian.

Output: The weight matrix of SPLRR based graph $\mathbf{Z}$.

Normalize all the samples $\mathbf{x}_i$ to ℓ^2 unit norm.

Solve problem (8) using Algorithm 1. and get the optimal solution $\mathbf{Z}^\star$.

Normalize each column of $\mathbf{Z}^\star$ via $\mathbf{z}_i^\star = \mathbf{z}_i^\star/||\mathbf{z}_i^\star||_2$ and shrink entries in $\mathbf{Z}^\star$ by θ.

Construct the graph weight matrix $\mathbf{W}$ by $\mathbf{W} = \left(|\mathbf{Z}^\star| + (|\mathbf{Z}^\star|)^T\right)/2$.

4 Semi-supervised Classification

Denote $\mathbf{Y} = [(\mathbf{y}^1)^T; (\mathbf{y}^2)^T; \cdots; (\mathbf{y}^n)^T] \in \mathbb{R}^{n\times c}$ as the initial label matrix. If $\mathbf{x}_i$ is the unlabeled data, then $\mathbf{y}^i = \mathbf{0}$. If $\mathbf{x}_i$ is labeled data in class k, then the k-th element of $\mathbf{y}^i$ is 1 and the other elements of $\mathbf{y}^i$ are 0. Generally, graph based semi-supervised learning models solve the following problem [13]:

$$\min_{\mathbf{Q}} \mathrm{Tr}(\mathbf{Q}^T\tilde{\mathbf{L}}\mathbf{Q}) + \mathrm{Tr}((\mathbf{Q} - \mathbf{Y})^T\mathbf{U}(\mathbf{Q} - \mathbf{Y})), \tag{14}$$

where $\tilde{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ is the normalized graph Laplacian, $\mathbf{U}$ is a diagonal matrix with the i-th diagonal element to control the impact of the initial label $\mathbf{y}^i$ of $\mathbf{x}_i$, $\mathbf{Q} \in \mathbb{R}^{n\times c}$ is the label matrix to be solved. For fair comparison, we simply set $U_{ii} = 1$ for all algorithms in our experiments.

Taking the derivative of Eq.(14) w.r.t $\mathbf{Q}$ and setting it to zero, we have:

$$\mathbf{L}\mathbf{Q} + \mathbf{U}(\mathbf{Q} - \mathbf{Y}) = \mathbf{0} \Rightarrow \mathbf{Q} = (\mathbf{L} + \mathbf{U})^{-1}(\mathbf{U}\mathbf{Y}). \tag{15}$$

5 Experiments

ORL, Extended Yale B and CMU PIE data sets are used in our experiments.

• **ORL**: There are 10 gray scale images for each of the 40 subjects. They were taken at different times, varying the lighting, facial expressions and facial details. Each face image is cropped and resized to 32×32.
• **Extended Yale B**: This database has 38 individuals, each subject having around 64 near frontal images under different illuminations. We simply use the first 50 cropped images of the first 20 individuals, and then resize them to 32×32.
• **PIE**: It contains 41368 images of 68 subjects with different poses, illumination and expressions. We only use their images in five near frontal poses and under different illuminations and expressions. The first 50 images of the first 20 subjects are selected. Each image is manually cropped and resized to size 32×32.

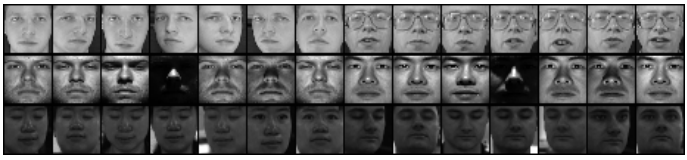


Fig. 1. Sample images from ORL, Extended Yale B and CMU PIE data sets

Some sample images from these three face databases are shown in Figure 1.

For evaluating the performance of proposed model, we compare SPLRR with some state-of-the-art graph construction models listed as follows:

• $k$nn-graph: The number of nearest neighbors for KNN1 and KNN2 are 4 and 8 respectively. The distance is measured using heat kernel and the kernel parameter is the average of squared Euclidean distances for all edged pairs.
• LNP [11]: We follow the pipeline of linear label propagation in to construct the graph. The neighborhood size in LNP is set to 40 to achieve the best results.
• ℓ^1 graph [5]: The ℓ^1 regularization item λ is empirically set to 0.01. And the ℓ^1 regularized least square problem is solved by $l1$-ls package.
• SPG graph [6]: we implement the SPG algorithm by setting n_{knn} as 10% of the size of data set and $\lambda = 0.001$.
• LRR graph [12]: the λ in LRR is set to an near optimal value 0.1.
• SPLRR graph: The number of nearest neighbors for 'local affinity' and 'distant repulsion' constraints are 4 and 20 respectively. The kernel parameter is empirically set as $0.1 \times \sqrt{-\bar{d}/\ln\{0.1/k\}}$ [9] ($\bar{d}$ is the average of squared Euclidean distances for all edged pairs on the graph, k is the neighbor number to construct the neighborhood graph). The hyper-parameters λ, α and β are empirically set as 0.5, 0.9 and 0.1 for all data sets.

For each face data set, we randomly select 10% to 60% face images per subject as labeled samples and the rest as unlabeled samples. Tables 1 reports the face recognition results on these three data sets.For each configuration, we conduct 50

Table 1. Experimental results on ORL, Extended Yale B and PIE(mean±std-dev%)

ORL	KNN1	KNN2	LNP	L1-Graph	SPG	LRR	SPLRR
10%	64.98±2.07	53.47±2.80	71.21±2.30	61.89±2.69	65.92±2.42	71.63±2.45	**77.31±2.17**
20%	74.07±2.71	63.97±2.69	82.01±2.20	76.16±2.73	78.74±1.98	83.67±2.12	**87.73±2.17**
30%	79.42±2.24	68.93±2.72	88.06±2.45	84.61±2.32	86.24±2.36	88.66±1.79	**91.66±1.67**
40%	81.16±2.43	71.64±2.82	91.43±1.68	89.31±1.90	90.72±1.64	91.35±1.72	**94.11±1.84**
50%	82.76±2.41	72.83±2.44	93.15±1.74	92.47±1.69	93.20±1.64	93.54±1.41	**95.93±1.51**
60%	83.23±2.20	74.75±2.69	94.69±1.64	94.25±1.59	95.46±1.50	94.46±1.55	**96.94±1.38**
YaleB	KNN1	KNN2	LNP	L1-Graph	SPG	LRR	SPLRR
10%	73.04±1.62	55.82±2.91	86.22±1.52	77.69±1.74	82.92±1.62	87.18±1.35	**92.96±0.98**
20%	77.13±1.31	63.03±2.39	90.86±1.14	87.58±1.05	89.84±1.16	92.36±1.17	**95.86±0.92**
30%	80.12±1.38	67.29±1.91	92.45±0.72	92.13±1.15	92.75±0.99	93.98±0.99	**97.31±0.65**
40%	81.49±1.35	69.86±2.46	93.42±0.86	94.39±0.94	94.26±0.77	95.42±0.81	**98.23±0.53**
50%	83.50±1.43	72.24±2.45	93.85±0.82	95.90±1.03	95.59±0.72	96.13±0.82	**98.75±0.42**
60%	84.25±2.03	74.74±2.42	94.89±0.98	97.29±0.93	96.37±0.85	96.73±0.86	**99.12±0.45**
PIE	KNN1	KNN2	LNP	L1-Graph	SPG	LRR	SPLRR
10%	49.38±2.83	40.09±2.11	67.50±2.77	63.60±2.35	65.29±2.16	75.93±2.05	**77.07±2.32**
20%	59.22±2.24	51.99±1.93	79.11±1.47	76.43±1.18	77.80±1.70	87.00±1.42	**88.11±1.65**
30%	64.69±1.73	60.76±1.77	83.54±1.73	82.93±1.44	83.82±1.26	90.33±1.17	**91.20±1.23**
40%	67.12±1.91	68.56±1.45	87.02±1.35	86.99±1.25	87.23±1.25	92.39±1.05	**93.56±1.09**
50%	69.87±2.09	75.11±1.10	89.08±1.38	89.34±1.02	89.35±1.41	93.79±1.11	**94.60±1.04**
60%	71.49±2.04	81.15±1.11	90.07±1.53	90.96±1.32	91.60±1.63	94.79±1.02	**95.47±1.34**

independent runs for each algorithm. The mean accuracy as well as the standard deviation of the performance are reported.

From the experimental results, we can observe that: 1) The LRR based methods, both LRR and SPLRR, perform consistently well on three data sets, which suggests that the LRR induced affinity matrix is efficient for semi-supervised learning. Generally, LNP, L1-Graph and SPG have similar performances while the baseline KNN graph has the lowest accuracy. 2) SPLRR graph consistently achieves the lowest classification error rates over other graphs even with low labeling percentages which can be observed from results on ORL and Extended Yale B data sets. When there are only 10% labeled samples, SPLRR can still obtain very high accuracies. This means the structure information of data set, which should be preserved in transformation process, is important when building an informative graph for semi-supervised learning.

6 Conclusion

In this paper, we have proposed a new graph construction model for semi-supervised face recognition, called structural preserving low-rank representation (SPLRR). SPLRR constructs the graph with preserving the structure of data set, which enforces the local affinity property to be preserved without distorting of the distant repulsion property. As a result, the proposed model derives

a informative graph and shows the best performance in the comparison with state-of-the-art methods for semi-supervised face recognition.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China (Grant No.61272248), the National Basic Research Program of China (Grant No.2013CB329401) and the Science and Technology Commission of Shanghai Municipality (Grant No.13511500200).

References

1. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized nonnegative matrix factorization for data representation. IEEE TPAMI 33(8), 1548–1560 (2011)
2. Cai, J.F., Candès, E.J., Shen, Z.: A singular value thresholding algorithm for matrix completion. SIAM J. Optimiz. 20(4), 1956–1982 (2010)
3. Candès, E.J., Li, X., Ma, Y., Wright, J.: Robust principal component analysis? J. ACM 58(3) (2011)
4. Carreira-Perpinán, M.A.: The elastic embedding algorithm for dimensionality reduction. In: ICML, pp. 167–174 (2010)
5. Cheng, B., Yang, J., Yan, S., Fu, Y., Huang, T.S.: Learning with ℓ^1-graph for image analysis. IEEE TIP 19(4), 858–866 (2010)
6. He, R., Zheng, W.S., Hu, B.G., Kong, X.W.: Nonnegative sparse coding for discriminative semi-supervised learning. In: CVPR, pp. 2849–2856 (2011)
7. Lin, Z., Chen, M., Ma, Y.: The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. arXiv preprint arXiv:1009.5055 (2010)
8. Liu, G., Lin, Z., Yu, Y.: Robust subspace segmentation by low-rank representation. In: ICML, vol. 3 (2010)
9. Nie, F., Xiang, S., Jia, Y., Zhang, C.: Semi-supervised orthogonal discriminant analysis via label propagation. Pattern Recogn. 42(11), 2615–2627 (2009)
10. Tenenbaum, J.B., De Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
11. Wang, F., Zhang, C.: Label propagation through linear neighborhoods. IEEE TKDE 20(1), 55–67 (2008)
12. Yang, S., Wang, X., Wang, M., Han, Y., Jiao, L.: Semi-supervised low-rank representation graph for pattern recognition. IET IP 7(2), 131–136 (2013)
13. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NIPS, vol. 16 (2004)

Marginalized Denoising Autoencoder via Graph Regularization for Domain Adaptation

Yong Peng[1], Shen Wang[2], and Bao-Liang Lu[1,3,⋆]

[1] Center for Brain-Like Computing and Machine Intelligence,
Department of Computer Science and Engineering,
Shanghai Jiao Tong Unviersity, Shanghai 200240 China
[2] Department of Electrical Engineering and Computer Science,
University of Michigan, Ann Arbor, 48109 USA
[3] MoE-Microsoft Key Lab. for Intelligent Computing and Intelligent Systems,
Shanghai Jiao Tong Unviersity, Shanghai 200240 China
bllu@sjtu.edu.cn

Abstract. Domain adaptation, which aims to learn domain-invariant features for sentiment classification, has received increasing attention. The underlying rationality of domain adaptation is that the involved domains share some common latent factors. Recently neural network based on Stacked Denoising Auto-Encoders (SDA) and its marginalized version (mSDA) have shown promising results on learning domain-invariant features. To explicitly preserve the intrinsic structure of data, this paper proposes a marginalized Denoising Autoencoders via graph Regularization (GmSDA) in which the autoencoder based framework can learn more robust features with the help of newly incorporated graph regularization. The learned representations are fed into the sentiment classifiers and experiments show that the GmSDA can effectively improve the classification accuracy when comparing with some state-of-the-art models on the cropped Amazon benchmark data set.

Keywords: Domain Adaptation, Marginalized Denoising Autoencoder, Graph Regularization.

1 Introduction

Sentiment analysis [9] aims to determine the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document which is now a popular application; however, it often suffers from cross domain learning curse. To solve this problem, one solution is domain adaptation, which can build classifiers that are robust to mismatched distributions [1] [8] [12]. This presents a major difficult in adapting predictive models. Recent work has investigated techniques for alleviating the difference: instance re-weighting [8], sub-sampling from both domains [5] and learning joint target and source feature representations[4] [7] [12] [6].

⋆ Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 156–163, 2013.

Learning domain-invariant features is under the assumption that there is a domain invariant feature space [4] [2] [3] where the source and target domains have the same or similar marginal distributions and the posterior distribution of the labels are the same across domains. A deep model (SDA: stacked denoising autoencoders) was proposed in [7] to learn domain-invariant features. Denoising autoencoder [10] is a single layer neural network, whose output aims at reconstructing the partially corrupted input. Denoizers can be used as building block to construct deep architecture. The linearized version of SDA: marginalized stacked denoising autoencoder (mSDA)[6], is computational economy with a closed form solution and has few hyper-parameters to tune.

In real applications, the data is likely to reside on a low-dimensional ambient space. It has been shown that the geometrical information of the data is important for pattern recognition. Though deep model has show promising performance on domain adaption, it does not explicitly considers the intrinsic structure of data. To compensate this drawback and simultaneously harness the great power of feature learning of deep architecture, we propose a graph regularized marginalized SDA, which considers the local manifold structure of the data. The graph regularization term can be seen as a smooth operator for making the learned features vary smoothly along the geodesics of the data manifold.

The remainder of this paper is organized as follows. Brief review on mSDA is given in section 2. The proposed model, marginalized Denoising Autoencoders via graph Regularization (GmSDA), is introduced in section 3. Section 4 evaluates our method on a benchmark composed of reviews of 4 types of Amazon products and section 5 is conclusion.

Notation and Background. We assume the data originates from two domains, source S and target T. We samples data $\mathcal{D}_{\mathcal{S}} = \{\mathbf{x}_1, \cdots, \mathbf{x}_{n_S}\} \in \mathbb{R}^d$ with ground truth label $L_S = \{y_1, \cdots, y_{n_S}\}$. For target domain, only data without labels $\mathcal{D}_{\mathcal{T}} = \{\mathbf{x}_{n_S+1}, \cdots, \mathbf{x}_n\} \in \mathbb{R}^d$ are available. We do not assume that both use identical features and pad all input vectors with zeros to make both domain have same dimensionality d. The goal is to learn a classifier $h \in \mathcal{H}$ with labeled data $\mathcal{D}_{\mathcal{S}}$ and unlabeled data $\mathcal{D}_{\mathcal{T}}$ to predict labels T of data in $\mathcal{D}_{\mathcal{T}}$.

2 Marginalized Denoising Autoencoders

mSDA is a linearized version of SDA, in which the building block of mSDA is a single layer denoising autoencoder. Given data points $\mathcal{D} = \{\mathbf{x}_1, \cdots, \mathbf{x}_U\} \in \mathbb{R}^d$, where $\mathcal{D} = \mathcal{D}_{\mathcal{S}} \bigcup \mathcal{D}_{\mathcal{T}}$, corruption is applied to them by random feature removal. Then each feature has a probability p to be set to 0. Denote the corrupted version of $\mathbf{x}_i$ as $\tilde{\mathbf{x}}_i$. Reconstruction of corrupted input using mapping $\mathbf{W} : \mathbb{R}^d \to \mathbb{R}^d$ is equal to minimizing the squared reconstruction loss:

$$\frac{1}{2n} \sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{W}\tilde{\mathbf{x}}_i||^2. \tag{1}$$

We can incorporate the bias into the mapping $\mathbf{W} = [\mathbf{W}, \mathbf{b}]$ with slightly modifying the feature as $\mathbf{x}_i = [\mathbf{x}_i; 1]$. And we assume that the constant feature is

never corrupted. Considering a low variance, the m times passes over the input are implied to corrupt different feature each time. Then the problem becomes to solve the $\mathbf{W}$ which aims to minimize the overall squared loss:

$$\frac{1}{mn}\sum_{j=1}^{m}\sum_{i=1}^{n}||\mathbf{x}_i - \mathbf{W}\tilde{\mathbf{x}}||^2. \tag{2}$$

By defining the design matrix $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \in \mathbb{R}^{d\times n}$, its m-times repeated version as $\overline{\mathbf{X}} = [\mathbf{X}, \cdots, \mathbf{X}]$ and corrupted version of $\overline{\mathbf{X}}$ as $\tilde{\mathbf{X}}$, (1) can be reduced to

$$\frac{1}{2mn}\mathrm{Tr}\left[(\overline{\mathbf{X}} - \mathbf{W}\tilde{\mathbf{X}})^T(\overline{\mathbf{X}} - \mathbf{W}\tilde{\mathbf{X}})\right], \tag{3}$$

whose solution can be expressed as the closed form solution for ordinary least squares:

$$\mathbf{W} = \mathbf{P}\mathbf{Q}^{-1} \quad \text{with} \quad \mathbf{Q} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T \quad \text{and} \quad \mathbf{P} = \bar{\mathbf{X}}\tilde{\mathbf{X}}^T. \tag{4}$$

Let $m \to \infty$, denoising transform $\mathbf{W}$ can be effectively computed with infinitely many copies of noise data. By the weak law of large numbers, $\mathbf{P}$ and $\mathbf{Q}$ converge to their mean values when $m \to \infty$. Then the mapping $\mathbf{W}$ can be expressed as:

$$\mathbf{W} = \mathbb{E}[\mathbf{P}]\mathbb{E}[\mathbf{Q}]^{-1} \quad \text{with} \quad \mathbb{E}[\mathbf{Q}] = \sum_{i=1}^{n}[\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T]. \tag{5}$$

Off-diagonal entries in $\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T$ are uncorrupted with the probability $(1-p)^2$, while for diagonal entries, this holds with probability $1-p$. Denote a vector $\mathbf{q} = [1-p, \cdots, 1-p]^T \in \mathbb{R}^{d+1}$, where $\mathbf{q}_\alpha$ and $\mathbf{q}_\beta$ represent the probabilities of no corruption happen to the feature α and β respectively. Defining the scatter matrix of the original uncorrupted input as $\mathbf{S} = \mathbf{X}\mathbf{X}^T$, the mean of $\mathbf{Q}$ can be expressed as:

$$\mathbb{E}[\mathbf{Q}]_{\alpha,\beta} = \begin{cases} \mathbf{S}_{\alpha\beta}\mathbf{q}_\alpha\mathbf{q}_\beta & \text{if } \alpha \neq \beta \\ \mathbf{S}_{\alpha\beta}\mathbf{q}_\alpha & \text{if } \alpha = \beta. \end{cases} \tag{6}$$

Similarly, the mean of $\mathbf{P}$ can be expressed as

$$\mathbb{E}[\mathbf{Q}]_{\alpha,\beta} = \mathbf{S}_{\alpha\beta}\mathbf{q}_\beta. \tag{7}$$

Then the reconstruction mapping $\mathbf{W}$ can be computed directly. This is the algorithm of the marginalized denoising autoencoder (mDA) [6].

Usually, the nonlinearity and the deep architecture is beneficial to feature learning. The nonlinearity is injected through the nonlinear quashing function $h(\cdot)$ after the reconstruction mapping $\mathbf{W}$ is computed. To perform the layer-wise stacking, several mDA layers are stacked by feeding the output of the $(t-1)$-th mDA (after the squashing function) as the input into the t-th layer mDA.

3 Marginalized SDA with Graph Regularization

In this section, we present our graph regularized marginalized Stacked Autoencoder (GmSDA) model.

3.1 General Graph Regularization Framework

As described in [11], a general class of graph regularization algorithms described by the following optimization problem: given the data $\mathcal{D} = \{\mathbf{x}_1, \cdots, \mathbf{x}_U\} \in \mathbb{R}^d$, we need to find a transformed representation $f(\mathbf{x}_i)$ w.r.t. $\mathbf{x}_i$ by minimizing

$$\sum_{i,j=1}^{U} \mathcal{L}(f(x_i, \alpha), f(x_j, \alpha), W_{ij}) \tag{8}$$

w.r.t. λ, subject to *Balance constraint*.

This type of optimization problem has the following main notations: $f(\mathbf{x}) \in \mathbb{R}^n$ is the embedding one trying to learn from a given example $\mathbf{x} \in \mathbb{R}^d$. It is parameterized by λ. In many techniques $f(\mathbf{x}_i) = f_i$ is a lookup table where each example i is assigned an independent vector f_i. $\mathcal{L}$ is a loss function between pairs of examples. Each element W_{ij} in $\mathbf{W}$ specifies the similarity or dissimilarity between samples $\mathbf{x}_i$ and $\mathbf{x}_j$. A balance constraint is often required for certain object functions so that a trivial solution is not reached.

3.2 Marginalized SDA with Graph Regularization

We propose the mSDA based deep learning system with graph regularization to learn domain-invariant features, which are used for training a linear SVM sentiment classifier. Our method can maximize the empirical likelihood (by DA) [10] and preserve the geometric structure (by graph regularization) simultaneously.

Considering a graph with N vertices where each vertex corresponds to a data point in the data set. The edge weight matrix $\mathbf{S}$ is usually defined as follows:

$$S_{ij} = \begin{cases} 1, & \text{if } \tilde{\mathbf{x}}_i \in \mathcal{N}_p(\tilde{\mathbf{x}}_j) \text{ or } \tilde{\mathbf{x}}_j \in \mathcal{N}_p(\tilde{\mathbf{x}}_i) \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

$\mathcal{N}_p(\mathbf{x}_i)$ denotes the set of p nearest neighbors of $\mathbf{x}_i$. Let f_i and f_j be the transformed representation (embedding) corresponding to $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_j$ respectively, where $f_i = \mathbf{W}\tilde{\mathbf{x}}_i$, $f_j = \mathbf{W}\tilde{\mathbf{x}}_j$, we hope to preserve the local structure of data by minimizing the following equation:

$$\frac{1}{2} \sum_{i,j=1}^{n} ||f_i - f_j||^2 S_{ij} = \mathrm{Tr}(\mathbf{W}\tilde{\mathbf{X}}\mathbf{L}\tilde{\mathbf{X}}^T\mathbf{W}^T), \tag{10}$$

where $\mathbf{L}$ is the graph Laplacian, which can be obtained by $\mathbf{L} = \mathbf{D} - \mathbf{S}$. $\mathbf{D}$ is a diagonal matrix whose entries are column (or row, since $\mathbf{S}$ is symmetric) sums of $\mathbf{S}$, $D_{ii} = \sum_j S_{ij}$.

By integrating this graph regularization term into the objective function of mDA, we can get the objective function of the building block for our model:

$$\arg\min_{\mathbf{W}} \frac{1}{2mn} \mathrm{Tr}\left[(\bar{\mathbf{X}} - \mathbf{W}\tilde{\mathbf{X}})^T(\overline{\mathbf{X}} - \mathbf{W}\tilde{\mathbf{X}})\right] + \mathrm{Tr}(\mathbf{W}\tilde{\mathbf{X}}\mathbf{L}\tilde{\mathbf{X}}^T\mathbf{W}^T), \tag{11}$$

which can be solved analytically

$$\mathbf{W} = \mathbf{P}(\mathbf{Q} + \lambda\tilde{\mathbf{X}}\mathbf{L}\tilde{\mathbf{X}}^T)^{-1}, \tag{12}$$

where $\mathbf{P}$ and $\mathbf{Q}$ have the same definition in Eq.(3) and λ represents the parameter to balance the contribution of the graphic regularization. Follow the marginalized configuration in Section 2. We can solved $\mathbf{W}$ in closed form as in Eq.(4). The whole process of our GmSDA model is summarized in Algorithm 1.

Algorithm 1. mSDA via Graph Regularization (GmSDA)

Input: Data point $\mathcal{D} = \{\mathbf{x}_1, \cdots, \mathbf{x}_U\} \in \mathbb{R}^d$, where $\mathcal{D} = \mathcal{D}_\mathcal{S} \bigcup \mathcal{D}_\mathcal{T}$, number of the layer l, corruption level p, number of nearest neighbors k, parameter λ to balance the contribution of the graph regularizer
Output: hidden representation of l layer $\mathbf{h}^l$
Construct a weighted graphic S by KNN in Binary style;
Compute the graph Laplacian $\mathbf{L}$;
Initialize $\mathbf{X}^0 = D$;
for $t \leftarrow 1$ ***to*** l **do**
  Compute $\tilde{\mathbf{X}}^{t-1}\tilde{\mathbf{X}}^{t-1T}$, $\overline{\mathbf{X}}^{t-1}\tilde{\mathbf{X}}^{t-1T}$ and $\tilde{\mathbf{X}}^{t-1}\mathbf{L}\tilde{\mathbf{X}}^{t-1T}$;
  Solve $\mathbf{W}^t$ according to Eq.(12);
  Compute $\mathbf{h}^t = \tanh(\mathbf{W}^t\mathbf{X}^{t-1})$;
  Define $\mathbf{X}^t = [\mathbf{X}^{t-1}; \mathbf{h}]$;
end
return $\mathbf{h}^l$

To apply GmSDA to domain adaptation, we first learn feature representation in an unsupervised fashion on the whole set including source domain and target domain data. Then the output of all layers, after squashing function $\texttt{tanh}(\mathbf{W}^t\mathbf{h}^{t-1})$, are combined with original features $\mathbf{h}_0$ to form new representations. Finally a linear SVM is trained on the new features.

4 Experiments

We evaluate GmSDA on the reduced *Amazon reviews* benchmark dataset [4] together with several other related algorithms. This data set is more controllable and contains review from 4 type of domains: books, DVDs, electronics and Kitchen appliances. For computational reasons, we followed the convention of [7] and [6], considering only binary classification problem: whether a review is positive or negative. The data is preprocessed as the setting in [4][6]. Our experiments are using the first 5000 features.

We followed the experimental configuration in [6]: training a linear SVM on the raw bag-of-words feature from the labeled source domain and test it on target domain as the baseline. PCA (as another baseline) is used to project the entire data set on to a low dimensional subspace where dense features are learned. Another three type of features are also used to train a linear SVM: structural correspondence learning (SCL) [4], 1-layers SDA [7] and mSDA[6].

We use *Transfer Loss*, *Transfer Ratio* and *Transfer Distance* [7] as metrics for evaluating the performance of models.

There are 4 parameters in GmSDA: the corruption level p, number of layers l, number of nearest neighbors k and the balance parameter λ. k and λ are set as 30 and 0.01 respectively in our experiments. p was selected with 5-fold cross validation on the labeled data on source domain, following the setup in [6]. Near optimal value p is obtained by this cross validation process for each domain.

Figure 1 displays the transfer loss across the twelve domain adaptation tasks. The GmSDA outperforms all the compared models, achieving the best performance. For some tasks, the transfer loss has negative results which denotes that the learned features from source domain can train a better classifier than the one trained on the original target domain. It is worth noticing that, GmSDA achieves a lower transfer loss in ten out of twelve tasks than mSDA, indicating that the learned features bridge the gap between domains.

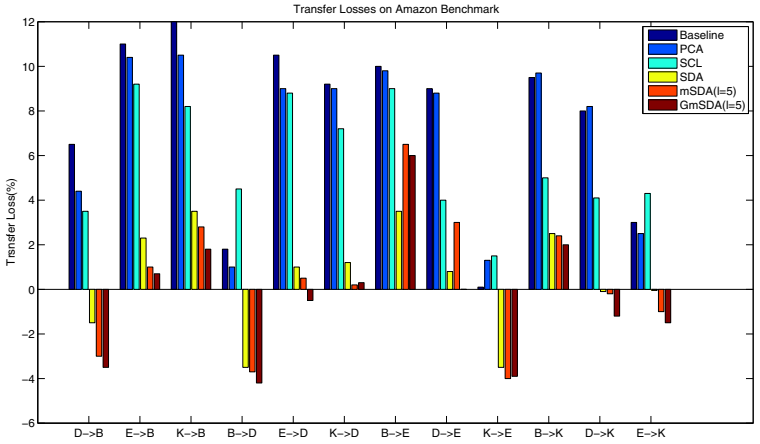


Fig. 1. Transfer losses on the Amazon benchmark of 4 domains: Books(B), DVDs(D), Electronics(E) and Kitchen(K) by different methods

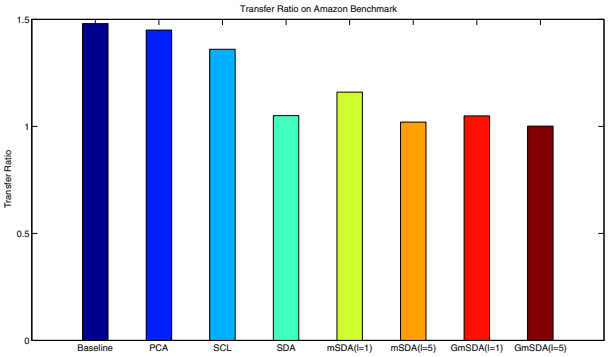


Fig. 2. Transfer ratios of algorithms on the Amazon benchmark

Figure 2 shows the transfer ratio for different methods and here we consider different layers of deep architectures as well. Compared with other methods, denoising autoencoder framework achieves better performance. In this framework, the deep architectures outperform the shallow ones and GmSDA get the best results. We can conclude that: 1). sharing the unsupervised pre-training across all domains is beneficial; 2). preserving the geometric structure is helpful to learn domain-invariant features; 3). deep architecture is better than the shallow one.

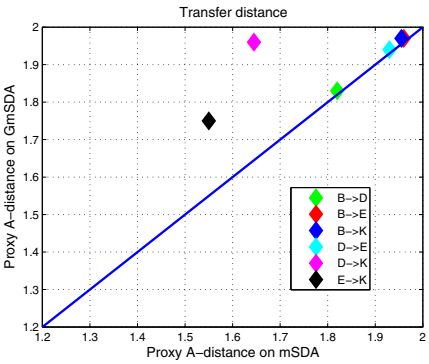


Fig. 3. Transfer distance: GmSDA vs. mSDA on the Amazon benchmark

Figure 3 shows the PAD of GmSDA and mSDA. All the points located beyond the blue line. It denotes that GmSDA features have bigger transfer distance than mSDA feature, which means it will be easier to distinguishing two domains with GmSDA features. We explain this effect through the fact that GmSDA is regularized with graph. With the help of the graph regularization, geometrical structure is exploited and the local invariance is considered, resulting a generally better representation. This helps both tasks, distinguishing between domains and sentiment analysis.

5 Conclusion

In this paper, we propose the mSDA based deep learning system with graph regularization. It can learn domain-invariant features which are suitable for sentiment classification. With help of the deep DA framework, we can maximize the empirical likelihood. Similarly, incorporating the graph regularization into mSDA, we can preserve the geometric structure to incorporate prior knowledge. This overcomes the shortcomings of most existing domain adaptation methods which focus only one aspect of the data or shallow framework. We compare our proposed approach against deep learning baselines over the reduced Amazon review benchmark. The experiments prove that our approach significantly outperforms all the baselines.

Acknowledgments. This work was supported partially by the National Natural Science Foundation of China (Grant No.61272248), the National Basic Research Program of China (Grant No.2013CB329401) and the Science and Technology Commission of Shanghai Municipality (Grant No.13511500200).

References

1. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Mach. Learn. 79(1-2), 151–175 (2010)
2. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In: ACL (2007)
3. Blitzer, J., Foster, D., Kakade, S.: Domain adaptation with coupled subspaces. In: AISTATS (2011)
4. Blitzer, J., McDonald, R., Pereira, F.: Domain adaptation with structural correspondence learning. In: EMNLP, pp. 120–128 (2006)
5. Chen, M., Weinberger, K.Q., Chen, Y.: Automatic feature decomposition for single view co-training. In: ICML (2011)
6. Chen, M., Xu, Z., Weinberger, K.Q., Sha, F.: Marginalized stacked denoising autoencoders. In: ICML (2012)
7. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: A deep learning approach. In: ICML (2011)
8. Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., Scholkopf, B.: Correcting sample selection bias by unlabeled data. In: NIPS (2007)
9. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Found. Trends Inform. Retrieval 2(1-2), 1–135 (2008)
10. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. 11, 3371–3408 (2010)
11. Weston, J., Ratle, F., Collobert, R.: Deep learning via semi-supervised embedding. In: ICML, pp. 1168–1175 (2008)
12. Xue, G.R., Dai, W., Yang, Q., Yu, Y.: Topic-bridged plsa for cross-domain text classification. In: SIGIR (2008)

Supporting Fuzzy-Rough Sets in the Dendritic Cell Algorithm Data Pre-processing Phase

Zeineb Chelly and Zied Elouedi

LARODEC, Institut Supérieur de Gestion de Tunis, Tunisia
zeinebchelly@yahoo.fr, zied.elouedi@gmx.fr

Abstract. The Dendritic Cell Algorithm (DCA) is an immune algorithm based on the behavior of dendritic cells. The DCA performance relies on its data pre-processing phase which includes two sub-steps; feature selection and signal categorization. For an automatic data pre-processing task, DCA applied Rough Set Theory (RST). Nevertheless, the developed rough approach presents an information loss as data should be discretized beforehand. Thus, the aim of this paper is to develop a new DCA feature selection and signal categorization method based on Fuzzy Rough Set Theory (FRST) which allows dealing with real-valued data with no data quantization beforehand. Results show that applying FRST, instead of RST, is more convenient for the DCA data pre-processing phase yielding much better performance in terms of accuracy.

Keywords: Evolutionary Algorithm, Fuzzy Rough Set Theory, Feature Selection.

1 Introduction

One of the emerging algorithms within the set of evolutionary computing algorithms is the Dendritic Cell Algorithm (DCA) [1]. DCA is inspired by the function of the natural dendritic cells. DCA has the ability to combine a series of informative signals with a sequence of repeating abstract identifiers, termed "antigens", to perform anomaly detection. To achieve this and through the pre-processing phase, DCA selects a subset of features and assigns each selected feature to a specific signal category; either as "Danger Signal" (DS), "Safe Signal" (SS) or as "Pathogen-Associated Molecular Pattern"(PAMP). The resulting correlation signal values are then classified to form an anomaly detection style of two-class classification. Technically and for an automatic DCA data pre-processing task, Rough Set Theory (RST) [2] was introduced leading to the development of various rough-DCA algorithms; namely RST-DCA [3], RC-DCA [4] and QR-DCA [5]. Based on the RST concepts and to perform feature selection, the rough DCA algorithms keep only a set of the most informative features, a subset termed *reduct*, that preserve nearly the same classification power of the original dataset; and to assign each selected feature to its specific signal category, the algorithms are based on the RST *reduct* and *core* concepts. The main difference between the algorithms is that RST-DCA assigns the same attribute

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 164–171, 2013.

to both SS and PAMP signals. However, RC-DCA and QR-DCA assign different attributes for SS and PAMP. Another main difference, is that QR-DCA looks for a trade-off between generating good classification results and preserving the lightweight of the DCA algorithm. More details about the algorithms can be found in [3][4][5]. Nevertheless, in all these crisp rough algorithms, the use of RST as a pre-processor technique is reliant upon crisp datasets; the feature values of the input dataset have to be discretized beforehand. Consequently, important information may be lost as a result of quantization [6]. This information loss may influence the rough-DCA algorithms, RST-DCA, RC-DCA, QR-DCA, feature selection process by generating an incorrect set of selected attributes; as a consequence, this will misguide the algorithms categorization phase by categorizing the features to erroneous signal categories. As a result, this will influence the algorithms classification process by generating unreliable classification results. To overcome the RST applicability restriction, Fuzzy Rough Set Theory (FRST) was introduced in [7] as it provides the means of data reduction for crisp and real-value attributed datasets which utilizes the extent to which values are similar. FRST encapsulates the related but distinct concepts of vagueness (for fuzzy sets) and indiscernibility (for rough sets), both of which occur as a result of uncertainty in data; a method employing fuzzy-rough sets can handle this uncertainty. Therefore, in this paper, we propose to develop a novel DCA version based on a new feature selection and signal categorization technique. Specifically, our new model, named FLA-DCA, is based on the framework of fuzzy rough set theory for data pre-processing and more precisely on the use of the fuzzy lower approximation (FLA); to guarantee a more rigorous data pre-processing phase. The main contributions of this paper are to introduce the concept of FRST in the DCA data pre-processing phase and to show how FRST can be applied to search for the convenient features to retain and how it can be appropriate for the categorization of each selected feature to its right type of signal. This will be achieved by avoiding the information loss already discussed and by keeping the attribute values unchanged with no need for a quantization process beforehand.

2 The Dendritic Cell Algorithm

DCA is a population based system, with each agent in the system is represented as a cell. Each cell has the capacity to collect data items, termed *antigens*. Formally, the DCA initial step is the automatic data pre-processing phase where feature selection and signal categorization are achieved. More precisely, DCA selects the most important features, from the initial input database, and assigns each selected attribute to its specific signal category (SS, DS or PAMP). Once data pre-processing is achieved and after calculating the values of the safe, PAMP and DS signals [1], DCA adheres these three signal categories and antigen to fix the context of each object (DC) which is the step of *Signal Processing*. In fact, the algorithm processes its input signals (already pre-categorized) in order to get three output signals: costimulation signal (Csm), semi-mature signal (Semi) and mature signal (Mat) [1]. A migration threshold is incorporated

into the DCA in order to determine the lifespan of a DC. As soon as the Csm exceeds the migration threshold; the DC ceases to sample signals and antigens. The migration state of a DC to the semi-mature state or to the mature state is determined by the comparison between cumulative $Semi$ and cumulative Mat. If the cumulative $Semi$ is greater than the cumulative Mat, then the DC goes to the semi-mature context, which implies that the antigen data was collected under normal conditions. Otherwise, the DC goes to the mature context, signifying a potentially anomalous data item. This step is known to be the *Context Assessment* phase. The nature of the response is determined by measuring the number of DCs that are fully mature and is represented by the Mature Context Antigen Value (MCAV). $MCAV$ is applied in the DCA final step which is the *Classification* procedure and used to assess the degree of anomaly of a given antigen. The closer the $MCAV$ is to 1, the greater the probability that the antigen is anomalous. By applying thresholds at various levels, analysis can be performed to assess the anomaly detection capabilities of the algorithm. Those antigens whose $MCAV$ are greater than the anomalous threshold, which can be automatically generated from the input data, are classified as anomalous while the others are classified as normal.

3 Fuzzy-Rough Sets for Feature Selection

1) Basic Concepts: In the same way that crisp equivalence classes are central to rough sets [2], *fuzzy equivalence classes* are central to the fuzzy-rough set approach [7]. For typical applications, this means that the decision values and the conditional values may all be fuzzy. The concept of crisp equivalence classes can be extended by the inclusion of a fuzzy similarity relation S on the universe, which determines the extent to which two elements are similar in S. The fuzzy lower and fuzzy upper approximations become $\mu_{\underline{R_P}X}(x) = inf_{y\in U} I(\mu_{R_P}(x,y), \mu_X(y))$ and $\mu_{\overline{R_P}X}(x) = sup_{y\in U} T(\mu_{R_P}(x,y), \mu_X(y))$. In the presented formulae, I is a fuzzy implicator and T is a t-norm. R_P is the fuzzy similarity relation induced by the subset of features P: $\mu_{R_P}(x,y) = \bigcup_{a\in P}\{\mu_{R_a}(x,y)\}$ where $\mu_{R_a}(x,y)$ is the degree to which objects x and y are similar for feature a. A fuzzy similarity relation can be constructed for this purpose, defined as: $\mu_{R_a}(x,y) = max(min(\frac{(a(y)-(a(x)-\sigma_a))}{(a(x)-(a(x)-\sigma_a))}, \frac{((a(x)+\sigma_a)-a(y))}{((a(x)+\sigma_a)-a(x))}), 0)$ where σ_a is the standard deviation of feature a. The fuzzy lower approximation contains information regarding the extent of certainty of object membership to a given concept. The fuzzy upper approximation contains information regarding the degree of uncertainty of objects. The couple $< \underline{P}(X), \overline{P}(X) >$ is called a *fuzzy-rough set*.

2) Reduction Process: To search for the optimal subset of features, the fuzzy-rough reduct, the fuzzy positive region has to be calculated. Formally, in the traditional RST, the crisp positive region is defined as the union of the lower approximations. By the extension to the fuzzy principal, the membership of an object $x \in U$ belonging to the *fuzzy positive region* can be defined by: $\mu_{POS_{R_P(Q)}}(x) = sup_{X\in U/Q} \mu_{\underline{R_P}X}(x)$. Object x will not belong to the fuzzy positive region only if the fuzzy equivalence class it belongs to is not

a constituent of the fuzzy positive region. Using the definition of the fuzzy positive region, the *fuzzy-rough dependency function* can be defined as follows: $\gamma'_P(Q) = \frac{\sum_{x \in U} \mu_{POS_{R_P(Q)}(x)}}{|U|}$. As with crisp rough sets, the dependency of Q on P is the proportion of objects that are discernible out of the entire dataset. In the present approach, this corresponds to determining the fuzzy cardinality of $\mu_{POS_{R_P(Q)}}(x)$ divided by the total number of objects in the universe. A Fuzzy-Rough QuickReduct algorithm can be constructed for locating a fuzzy-rough reduct based on this measure. According to Fuzzy-Rough QuickReduct algorithm, the fuzzy dependency degree of the addition of each attribute to the current fuzzy reduct candidate, R, (initially empty) is calculated, and the best candidate is chosen. This process continues until the fuzzy dependency of the subset equals the fuzzy dependency degree (consistency) of the entire dataset, i.e., $\gamma'_R(D) = \gamma'_C(D)$. A worked example on how to compute a fuzzy-rough reduct using the Fuzzy-Rough QuickReduct algorithm, based on the fuzzy lower approximation, can be found in [6].

4 FLA-DCA: The Solution Approach

4.1 The FLA-DCA Signal Selection Process

For antigen classification, our learning problem has to select high discriminating features from the original input database which corresponds to the antigen information dataset. We may formalize this problem as an information table, where universe $U = \{x_1, x_2, \ldots, x_N\}$ is a set of antigen identifiers, the conditional attribute set $C = \{c_1, c_2, \ldots, c_A\}$ contains each feature of the information table to select and the decision attribute D of our learning problem corresponds to the class label of each sample. As FLA-DCA is based on the standard DCA concepts, except for the data pre-processing phase, and since DCA is applied to binary classification problems; then our developed FLA-DCA will be, also, applied to two-class datasets. Therefore, the decision attribute, D, of the input database of our FLA-DCA has binary values d_k: either the antigen is collected under safe circumstances reflecting a normal behavior (classified as normal) or the antigen is collected under dangerous circumstances reflecting an anomalous behavior (classified as anomalous). The condition attribute feature D is defined as follows: $D = \{normal, anomalous\}$. For feature selection, FLA-DCA computes, first of all, the fuzzy lower approximations of the two decision concepts d_k, for all attributes c_i and for all objects x_j; denoted by $\mu_{\underline{R_{c_i}}\{d_k\}}(x_j)$. Using these results, FLA-DCA calculates the fuzzy positive regions for all c_i, for each object x_j, defined as $\mu_{POS_{R_{c_i}}(D)}(x_j)$. Based on these calculations and to find the fuzzy-rough reduct, FLA-DCA starts off with an empty set and moves to calculate the fuzzy dependency degrees of D on c_i, defined as $\gamma'_{c_i}(D)$; as presented in Section 3. The attribute c_m having the greatest value of fuzzy-rough dependency degree is added to the empty fuzzy-rough reduct set. Once the first attribute c_m is selected, FLA-DCA adds, in turn, one attribute to the selected first attribute and computes the fuzzy-rough dependency degree of each obtained attributes'

couple $\gamma'_{\{c_m,c_i\}}(D)$. The algorithm chooses the couple having the greatest fuzzy-rough dependency degree. The process of adding each time one attribute to the subset of the selected features continues until the fuzzy-rough dependency degree of the obtained set of features equals the fuzzy-rough dependency degree, $\gamma'_C(D)$, of the entire database.

4.2 The FLA-DCA Signal Categorization Process

Our method has to assign, now, for each selected attribute, included in the generated fuzzy-rough reduct, its definite and specific signal category; either as SS, as DS or as a PAMP signal. The general guidelines for signal categorization are based on the semantic of each signal category:

Safe signals : They certainly indicate that no anomalies are present.
PAMPs : They usually means that there is an anomalous situation.
Danger signals : They may or may not show an anomalous situation, however the probability of an anomaly is higher than under normal circumstances.

Based on the immunological definitions stated above, it is clear that both PAMP and SS are positive indicators of an anomalous and normal signal while the DS is measuring situations where the risk of anomalousness is high, but there is no signature of a specific cause. This problem can be formulated as follows: Based on the semantics of the mentioned signals, a ranking can be performed for these signals. More precisely, both SS and PAMP are more informative than DS which means that both of these signals can be seen as indispensable attributes; reflecting the first and the second ranking positions. To represent this level of importance, our method uses the first obtained couple of features through the fuzzy-rough reduct generation. On the other hand, DS is less informative than PAMP and SS; reflecting the last and third ranking position. Therefore, our method applies the rest of the fuzzy-rough reduct attributes, discarding the two first selected attributes that are chosen to represent the SS and PAMP signals, to represent the DS. More precisely, our method processes as follows:

As FLA-DCA has already calculated the fuzzy-rough dependency degree of each attribute c_i a part, $\gamma'_{c_i}(D)$, FLA-DCA selects the first attribute c_m having the greatest fuzzy-rough dependency degree to form the SS as it is considered the most informative first feature added to the fuzzy-rough reduct set. With no additional computations and since FLA-DCA has already computed the fuzzy-rough dependency degrees of each attributes' couple $\gamma'_{\{c_m,c_i\}}(D)$ when adding, in turn, one attribute c_i to the selected first attribute c_m that represents the SS, FLA-DCA chooses the couple having the greatest dependency degree. More precisely, FLA-DCA selects that second attribute c_r having the greatest $\gamma'_{\{c_m,c_r\}}(D)$ among the calculated $\gamma'_{\{c_m,c_i\}}(D)$; to form the PAMP signal. Finally, the rest of the reduct attributes are combined and affected to represent the DS as it is less than certain to be anomalous. Once signal categorization is achieved, FLA-DCA processes its next steps as the DCA does [1].

5 Experimental Setup

To test the validity of our FLA-DCA fuzzy-rough model, our experiments are performed on two-class, real-valued attributes, databases from [8]. In [5], a comparison between the rough DCA algorithms is performed and it was shown that QR-DCA outperforms both RST-DCA and RC-DCA in terms of classification accuracy. Another characteristic of QR-DCA is that it takes less time to process than RC-DCA and RST-DCA. Therefore, in what follows we will compare our new fuzzy-rough developed approach, FLA-DCA, to the QR-DCA crisp rough algorithm.

Table 1. Description of Databases

Database	Ref	♯ Instances	♯ Attributes
Sonar	SN	208	61
Molecular-Bio	Bio	106	59
Spambase	SP	4601	58
Cylinder Bands	CylB	540	40
Ionosphere	IONO	351	35
Sick	Sck	3772	30

For data pre-processing, FLA-DCA and QR-DCA uses FRST and RST, respectively. For both approaches, each data item is mapped as an antigen, with the value of the antigen equal to the data ID of the item. For all DCA algorithms, a population of 100 cells is used. To perform anomaly detection, a threshold which is automatically generated from the data is applied to the MCAVs. The MCAV threshold is derived from the proportion of anomalous data instances of the whole dataset. Items below the threshold are classified as class one and above as class two. The resulting classified antigens are compared to the labels given in the original datasets. For each experiment, the results presented are based on mean MCAV values generated across a 10-fold cross validation. We evaluate the performance of the mentioned DCA methods in terms of number of extracted features, running time, sensitivity, specificity and accuracy which are defined as: $Sensitivity = TP/(TP + FN)$; $Specificity = TN/(TN + FP)$; $Accuracy = (TP+TN)/(TP+TN+FN+FP)$; where TP, FP, TN, and FN refer respectively to: true positive, false positive, true negative and false negative.

6 Results and Analysis

In this Section, we aim to show that applying FRST, instead of RST, can avoid the information loss caused by the mandatory step of data quantization. Furthermore, we aim to show that by leaving the attribute values unchanged, our proposed fuzzy-rough FLA-DCA algorithm is able to select fewer features than the crisp rough QR-DCA approach, leading to better guide the FLA-DCA algorithm classification task. This is confirmed by the results presented in Table 2. For instance, from Table 2, we can notice that our new fuzzy-rough DCA model,

Table 2. Comparison Results of the Rough DCA Approaches

Database	Specificity(%)		Sensitivity(%)		Accuracy(%)		Time(s)		♯ Attributes	
	DCA		DCA		DCA		DCA		DCA	
	QR	FLA	QR	FLA	QR	FLA	QR	FLA	QR	FLA
SN	92.79	96.90	89.19	95.49	90.86	96.15	7.79	9.41	22	10
Bio	79.24	92.45	77.35	86.79	78.30	89.62	5.25	8.47	19	9
SP	98.67	99.89	99.17	99.77	98.87	99.84	1976.05	2071.8	11	8
CylB	97.75	98.39	97.00	97.00	97.46	97.85	12.68	18.96	7	5
IONO	96.88	99.11	96.03	98.41	96.58	98.86	15.88	30.72	22	9
Sck	97.65	99.12	96.53	96.96	97.58	98.99	510.05	602.8	22	14

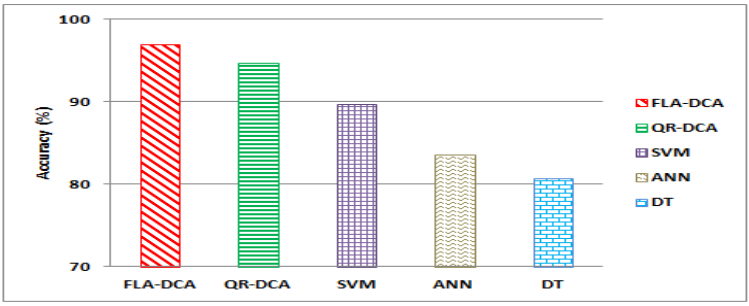


Fig. 1. Classifiers' Average Accuracies

FLA-DCA, selects fewer features than the crisp rough DCA model, QR-DCA. This is explained by the fact that FLA-DCA, by applying the Fuzzy-Rough QuickReduct algorithm, incorporates the information usually lost in crisp discretization by utilizing the fuzzy lower approximation to provide a more informed technique. For instance, applying FLA-DCA to the SN database, the number of selected attributes is reduced by more than 50% (10 features) in comparison to the number of features selected by QR-DCA, which is set to 22. Our FLA-DCA new approach performs much better than traditional RST on the whole, in terms of both feature selection and classification quality. For instance, when applying the algorithms to the SN dataset, the classification accuracy of FLA-DCA is set to 96.15%. However, when applying QR-DCA to the same database, the accuracy is set to 90.86%. Same remark is observed for the specificity and the sensitivity criteria. When comparing the results in terms of running time, we can notice that the time taken by our FLA-DCA to process is a bit longer than the time needed by QR-DCA to function. This is explained by the fact that our FLA-DCA incorporates the fuzzy component in comparison to QR-DCA.

We have, also, compared the performance of our FLA-DCA to other classifiers which are the Support Vector Machine (SVM), Artificial Neural Network (ANN) and the Decision Tree (DT). The comparison made is in terms of the average

of accuracies on the databases presented in Table 1. Fig.1 shows that that the highest classification accuracy is noticed for our fuzzy-rough DCA new model, FLA-DCA.

7 Conclusion and Future Work

A new hybrid DCA classification model based on fuzzy rough set theory is proposed in this paper. Our fuzzy-rough model ensures a more rigorous data pre-processing when dealing with datasets with real-valued attributes. Results show that our FLA-DCA is capable of performing better its classification task than the crisp rough model and other classifiers. As future work, we aim to boost the DCA data pre-processing phase by extending the application of FRST to databases with missing data.

References

1. Greensmith, J., Aickelin, U., Twycross, J.: Articulation and clarification of the dendritic cell algorithm. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 404–417. Springer, Heidelberg (2006)
2. Pawlak, Z.: Rough sets. International Journal of Computer and Information Science 11, 341–356 (1982)
3. Chelly, Z., Elouedi, Z.: RST-DCA: A dendritic cell algorithm based on rough set theory. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part III. LNCS, vol. 7665, pp. 480–487. Springer, Heidelberg (2012)
4. Chelly, Z., Elouedi, Z.: RC-DCA: A new feature selection and signal categorization technique for the dendritic cell algorithm based on rough set theory. In: Coello Coello, C.A., Greensmith, J., Krasnogor, N., Liò, P., Nicosia, G., Pavone, M. (eds.) ICARIS 2012. LNCS, vol. 7597, pp. 152–165. Springer, Heidelberg (2012)
5. Chelly, Z., Elouedi, Z.: QR-DCA: A new rough data pre-processing approach for the dendritic cell algorithm. In: Tomassini, M., Antonioni, A., Daolio, F., Buesser, P. (eds.) ICANNGA 2013. LNCS, vol. 7824, pp. 140–150. Springer, Heidelberg (2013)
6. Jensen, R., Shen, Q.: New approaches to fuzzy-rough feature selection. IEEE Transactions on Fuzzy Systems 17, 824–838 (2009)
7. Dubois, D., Prade, H.: Putting rough sets and fuzzy sets together. Kluwer Academic Publishers, Dordrecht (1992)
8. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007), `http://mlearn.ics.uci.edu/mlrepository.html`

Improving the Efficiency of Spiking Network Learning

Vaenthan Thiruvarudchelvan, Wayne Moore, and Michael Antolovich

Charles Sturt University, Bathurst, Australia
{vthiru,mantolovich}@csu.edu.au, wmoore@lisp.com.au

Abstract. Spiking networks are third generation artificial neural networks with a higher level of biological realism. This realism comes at the cost of extra computation, which alongside their complexity makes them impractical for general machine-learning applications. We propose that for some problems, spiking networks can actually be *more* efficient than second generation networks. This paper presents several enhancements to the supervised learning algorithm SpikeProp, including reduced precision, fewer subconnections, a lookup table and event-driven computation. The cputime required by our new algorithm SpikeProp+ was measured and compared to multilayer perceptron backpropagation. We found SpikeProp+ to use 20 times less CPU than SpikeProp for learning a classifier, but it remains ten times slower than the perceptron network. Our new networks are not optimal however, and several avenues exist for achieving further gains. Our results suggest it may be possible to build highly-efficient neural networks in this way.

Keywords: spiking neural networks, computational costs, efficiency.

1 Introduction

Within the vast field of neural networks, the term itself is most commonly associated with the sigmoid multilayer perceptron (sometimes called a second generation network), and it is commonplace in today's applications. Among other branches of the field are spiking neural networks, which some consider to be the third generation. Spiking networks simulate real neurons rather then emulate them, computing membrane potentials through time. Maass has shown their greater level of realism confers a greater representational ability [1].

As with simpler networks, there are numerous ways to make use of spiking networks. Examples of learning rules discovered thus far include SpikeProp, SpikeNet, ReSuMe and Tempotron. SpikeProp [2] is a general-purpose supervised machine learning rule whose behaviour is well understood after a decade's scrutiny. It uses the principle of error backpropagation first applied to the multilayer perceptron (Backprop) [3], using the same feedforward structure but working with spike times instead of abstract "activations".

SpikeProp is time-stepped, meaning neuron states are calculated at small time intervals called timesteps. Smaller timesteps better approximate continuous

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 172–179, 2013.

neuron dynamics, however this comes with a computation penalty – halving the timestep doubles the computation. Event-driven computation [4] and lookup tables [5] are two ways researchers have tackled this problem. Spiking algorithms are also more complex and entail many more parameters than Backprop. For these reasons, spiking networks have not made inroads in the general machine-learning community.

We propose that the heavy computational penalty of spiking networks can be eliminated, giving way to high-efficiency artificial neural networks. Our reasoning is that the brain's networks are highly adapted to achieve unprecedented energy-efficiency [6], something which spiking networks by their nature could reproduce. Brain networks are relatively noisy and slow, yet are able to perform highly complex processing, invoking only a fraction of their neurons at any time. A more efficient alternative to established neural networks would be desirable in view of the growing mountain of modern data.

In our work, we found SpikeProp to require up to two orders of magnitude more processing time to achieve the same classification performance as Backprop. This paper introduces several enhancements to the algorithm to reduce that computational footprint, described in the next section. The section after outlines our experiments comparing the new SpikeProp+ to SpikeProp and Backprop, and the experimental environment in detail. In section 4 we present the results, which are analysed in the Discussion along with their implications.

2 Methodology

A very brief description of SpikeProp is given here, details may be found in the original paper [2]. Networks take the familiar input-hidden-output layer feedforward structure, with each layer fully connected to the next. Each connection is composed of n subconnections or synapses, each with a weight and a progressive delay. A spike emitted from a neuron evokes at each target neuron a post-synaptic potential (PSP), scaled and delayed by each synapse. The PSP takes the form of an alpha function in the original algorithm, but others have been proposed [7]. The contributions of all PSPs are summed at the target neuron at every timestep, causing it to emit its own spike when a threshold is crossed. Neuronal dynamics after the first spike are ignored, since only its timing is used by the algorithm.

Training inputs are preprocessed into spike times for the input neurons, and the network is run for as many timesteps as required for all neurons to have spiked. The difference between output spike times and suitably preprocessed training values are squared and summed. Every weight is then adjusted in the direction minimising this sum-squared-error. The process is repeated until convergence in a manner essentially the same as Backprop [3].

2.1 Speeding Up SpikeProp

Here we describe the enhancements to SpikeProp used in this paper, which together comprise SpikeProp+. Shorthands used in the figures are in brackets.

Alternate spike response functions (int-, tanh-). In earlier work we discovered that using the spike response functions (SRFs) intalpha and tanh resulted in quicker and more reliable convergence over the standard (alpha) [7]. Here we find that they also convey computational benefits.

Unitary subconnections (-1). SpikeProp was intended to use a number of synapses per link between neurons. We found that with an appropriate time constant, a single subconnection performs just as well.

Lookup table (L). Many relatively slow floating point calculations can be avoided by pre-calculating and storing the value of the spike response function at every timestep over a fixed range (1000 steps here). These are stored in an array for rapid lookup [5].

Winner-takes-all optimisation (W). We focus on classification in this paper, with the output class indicated by the first output neuron to spike. Once that has occurred, no further computation of the network is required. This is only possible during the testing phase, since all output times are required by the SpikeProp algorithm.

Event-driven simulation (E). Instead of calculating the sum of PSPs at every every timestep, we can do this at a smaller number of *event* times. When a neuron spikes, events indicating the future arrival of PSPs are entered into a priority queue. The algorithm repeatedly removes the earliest event and calculates the membrane voltage at that time. If superthreshold, a spike is recorded, enqueuing further PSP events. If subthreshold, a second NEURON type event is enqueued, with a linear prediction of the spiking timestep. As long as the derivative of the SRF is monotonically decreasing (tanh only), the true first spiking timestep will not be missed, however multiple events will occur as the prediction is iteratively refined.

Coarse Timesteps (Sn). We found that despite the distorted dynamics, SpikeProp continues to function normally at timesteps as high as 1ms, depending on the dataset. Our experience suggests that the limiting factor is the ability to alias input patterns to separate timesteps.

Reducing convergence-testing (Tn). Since running a SpikeProp network is relatively expensive compared to Backprop, we can test for convergence once every n epochs while training. Testing half as often is unlikely to cause networks in training to fail when they otherwise wouldn't.

2.2 Measuring CPU Loading

When comparing networks of the same type, the number of epochs (one pass over every training record) is commonly used to compare learning performance. However, across algorithms this figure is not directly comparable because the amount of computation per epoch can vary significantly. This discrepancy is seen in [2], where SpikeProp needs three orders of magnitude fewer epochs than Backprop while it would be expected to run more slowly.

To compare the two algorithms, we measured the cputime of each training and testing run. Cputime is the amount of time a process is scheduled on a processor core by the OS, which is mostly independent of other running tasks.

It gives an accurate indication of the computational load of a program, which is closely related to the energy it consumes [8]. Cputime is a better measure than runtime, which can be influenced by other running tasks and multiprocessing. The term "speed" refers to runtime, but we use all three terms loosely to refer to computation.

Both SpikeProp and Backprop were implemented in the same program using object-oriented C++. This was done to eliminate as many runtime environment effects as possible, and by keeping the same network representation, observe only algorithmic differences. All experiments were run on a 3 GHz Intel Core 2 Duo CPU with 2GB RAM running Linux 3.5.0-31.

3 Experiments

Our experiments used two benchmark classification datasets, the Fisher Iris and Wisconsin Breast Cancer datasets [9] (table 1). All datapoints in the datasets used were scaled to [0..1] over the given ranges. These were then scaled to [-1..1] for Backprop inputs and outputs, for the tanh activation function. For SpikeProp, the datapoints were scaled and delayed to the intervals specified in table 2, and quantized to the nearest timestep. All networks included a reference input neuron whose value is always zero [10], and no biases were used. Backprop weights were randomly initialized in the interval [-2..2], while SpikeProp weights were initialized in [1..10] with thresholds equal to the number of incoming synapses ("method 6"). Network output decoding uses a winner-takes-all encoding where the earliest-firing (SpikeProp) or least-activated (Backprop) neuron codes for the desired class. Online learning (weights updated after every record rather than every epoch) was always used, for quicker learning [11]. Datasets were split in two by de-interleaving, one each for training and testing.

The first experiment was to find the optimal value of the learning rate η for each algorithm and best spike response function. These were indicated by the fewest epochs required (with acceptable failure rate) to reach 95% classification accuracy on the Iris dataset. Several values of learning rate were tried while SpikeProp's timestep was held at 0.1ms. Training was done for a maximum of 1000 epochs.

In the next experiment, we tested the efficacy of each SpikeProp enhancement described in the previous section, separately and in aggregate. Our event-driven algorithm requires the tanh SRF, so it was tried alongside intalpha. Timesteps of 0.1ms (if not mentioned) and 0.5ms were used. For unitary subconnections, we used the best coding intervals, time-constants and learning rates found through considerable parameter tuning (see table 2). For our final experiment, the two best SpikeProp+ configurations were compared to Backprop on both datasets.

4 Results

The results of our experiments are presented in figures 1 – 3. The plot on the left of each figure gives an idea of the convergence rate and success of each configuration. The epochs required for each run are marked with a cross, giving an idea

Table 1. Experimental datasets and network encoding. 16 records with missing data were removed from the Breast Cancer dataset, leaving 683 records.

Dataset	n Train	n Test	Missing	Attrs.	Outputs	Network	Neurons
Iris	75	75	0	4	3	5, 4, 3	12
WBC	342	341	16	9	2	10, 6, 2	18

Table 2. List of parameters used in experiments

Parameter	Value				Description
	alpha16	int16	int1	tanh1	
η	0.3	1	0.1	0.1	Learning rate
srf	alpha	intalpha	intalpha	tanh	Spike response function [7]
τ	7	1	7	21	SRF time-constant
ϑ	-				Spike threshold (see text)
ΔT_{in}	6	6	5	7	Input spike interval (ms)
ΔT_{out}	6	6	7	7	Output spike interval (ms)
T_{out}	10	10	7	7	Output spike delay (ms)
reference neuron	yes				Additional input with spike time 0
inhibitory neurons	0				Inhibitory hidden neuron count
subconnections	16	16	1	1	Synaptic terminals per connection
weight_init_method	6				Weight init. method (see text)
weight_random_seed	1–20				Random seed for weight init.
negative weights	yes				Permit negative weights
max_steps	5000				Max. steps per simulation
timestep	0.1, 0.5				Simulation step size (ms)
target	95% accuracy				Training target
max_epochs	1000				Epochs to wait for convergence

of their distribution, and the percentage of successful convergences are shown in the bar chart at the bottom (e.g. figure 1a). On the right, the cputimes elapsed for the training (red) and testing (green) phases are shown (e.g. figure 2b). Medians were taken to avoid outliers skewing the data, and error-bars indicate one standard-deviation. Plots were truncated to clearly display salient information.

The first experiment's results are shown in figure 1, highlighting the heavy computational cost of SpikeProp. Considering both the convergence data and cputimes, the best learning rates were selected: 0.1 for Backprop (cputime 0.036s) and 0.3, 1 and 1 for the SRFs alpha, intalpha and tanh respectively. With SpikeProp, intalpha is about twice as fast as the original alpha-function, tanh trailing at about 50% faster. We therefore focused on intalpha and tanh.

Figure 2 presents the improvements of each of the SpikeProp enhancements described in Section 2.1. (Testing cputime plots are overlaid on the training plots, not added). From left to right, we see again that with the original 16 subconnections, switching from alpha to intalpha roughly halves training time and it converges significantly better. Using intalpha hereafter, figure 2a shows that raising the timestep from 0.1ms to 0.5ms comes at a severe penalty in convergence, but improves training time slightly and more than halves the test

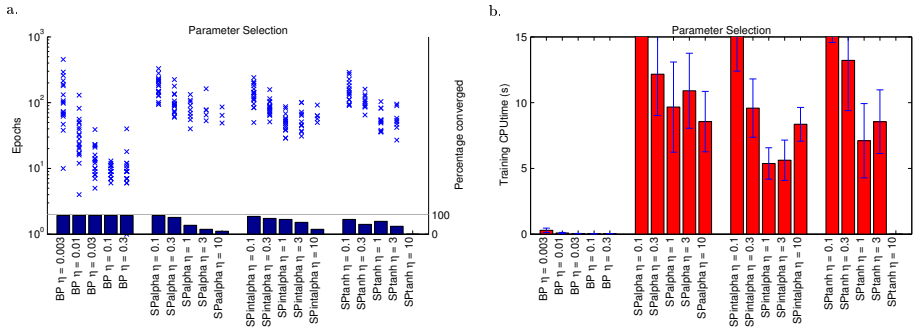


Fig. 1. Selecting the best η and SRF. SPtanh never converged for $\eta = 10$.

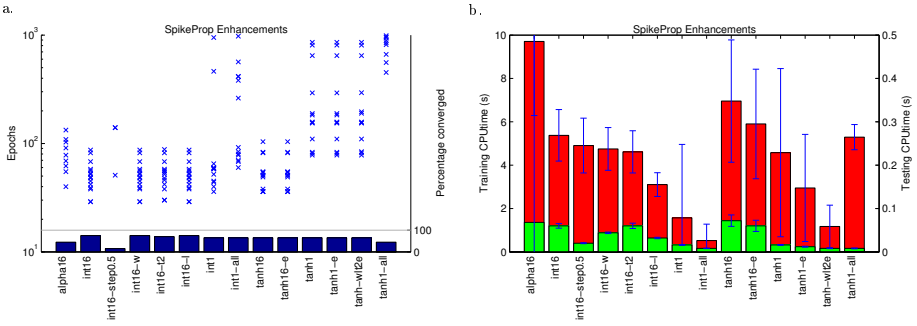


Fig. 2. Improvements to SpikeProp performance from each and all enhancements. (Note: int1-all is not event-driven, which requires the tanh function).

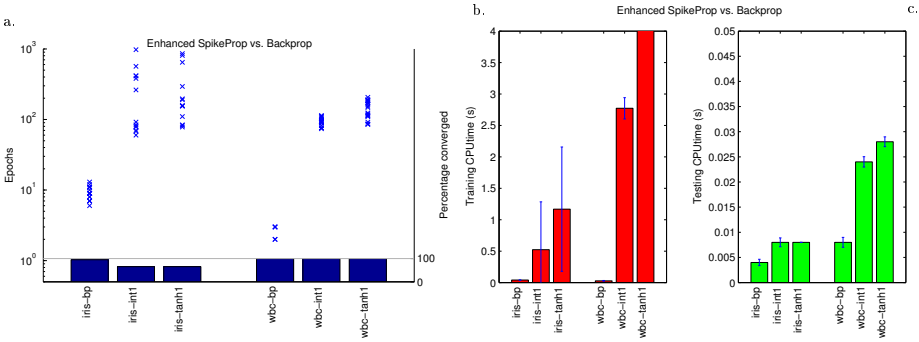


Fig. 3. Comparing the performance of our enhanced SpikeProp to Backprop

time (figure 2b). The WTA method produces a similar improvement in training time, but only about 25% improvement in testing time. Testing for convergence only every two epochs during training shows a roughly 10% benefit in training cputime, and obviously none for testing. Using a lookup table produces a significant 40% speedup in both training and testing. Finally, using a single subconnection rather than 16 gives a drastic speedup of around 70% to both training and testing. The int1-all setup more than halves this further, a mere 10% of the int16, and a minute 5% of the alpha16 setup originally described by Bohte et al. – a 20 times speedup. Notably, the earlier convergence penalty of the coarser timestep is not seen.

Since the event-driven algorithm currently only works with tanh, we conducted some further runs with it. With 16 subconnections, figure 2b shows the algorithm gives a 15% speed gain. Dropping to one subconnection however, the event-driven algorithm now takes 35% less training time and slightly improves testing time. The second-to-last run includes all enhancements with a timestep of 0.1ms, which is then raised to 0.5ms in the last run. The former is a mere 17% of the tanh16 setup, however this jumps over four times with the coarser timestep, along with a big drop in convergence. Evidently the coarser timestep prevents some of the enhancements from working with tanh. The best event-driven tanh run is still twice as slow as the best (non-event-driven) intalpha one.

In the final experiment, the two best setups from the previous experiment are compared against Backprop. Figure 3a shows a noticeable penalty in convergence for SpikeProp+ on the Iris dataset. In figure 3b, the intalpha version of SpikeProp+ has reduced to about 13 times Backprop in training, with the tanh version trailing at 29 times. For this dataset, SpikeProp+ testing cputimes are double that of Backprop but still tiny (figure 3c). On the WBC dataset, SpikeProp+ is orders of magnitude worse in all respects besides convergence. It is possible that parameter tuning needs to be repeated for the new dataset. All converged algorithms had similar test accuracies with minimal variance (90% Iris and 96% WBC).

5 Discussion

Our results show that SpikeProp will function several times faster with unitary subconnections. Alternate spike response functions and sacrificing precision with coarse timesteps also improve speed. SpikeProp performance was significantly improved with a lookup table, as did our event-driven scheme in certain conditions. Using these and other techniques in SpikeProp+, we were able to slash the computation levels of SpikeProp by a remarkable 20 times. In the best case, this still leaves SpikeProp+ about ten times slower than the multilayer perceptron trained with backpropagation.

We have good reason to believe that the remaining gap can be closed. Although considerable parameter tuning was needed to arrive at our best SpikeProp+ networks, they are unlikely to be optimal. The parameter space for SpikeProp is large, and more insightful selection could achieve further gains.

Our event-driven algorithm is also relatively inefficient; a better method would minimise iteration and use the superior intalpha spike response function. Furthermore, we have yet to implement improvements published by others [12,13], though ultimately another spiking algorithm may prove more efficient.

This work holds promise for building spiking neural networks with better computational efficiency than earlier networks. By importing properties like reduced-precision and event-based computation from the brain, spiking networks could take us a step towards that organ's efficiency. This is essentially bringing another level of biomimicry to artificial neural networks.

References

1. Maass, W.: Networks of spiking neurons: The third generation of neural network models. Neural Networks 10(9), 1659–1671 (1997)
2. Bohte, S.M., Kok, J.N., Poutré, H.L.: Error-backpropagation in temporally encoded networks of spiking neurons. Neurocomputing 48(1–4), 17–37 (2002)
3. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. tech. rep., Institute for Cognitive Science, University of California San Diego (September 1985)
4. Mattia, M., Giudice, P.D.: Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses. Neural Computation 12(10), 2305–2329 (2000)
5. Ros, E., Carrillo, R., Ortigosa, E.M., Barbour, B., Agís, R.: Event-driven simulation scheme for spiking neural networks using lookup tables to characterize neuronal dynamics. Neural Computation 18, 2959–2993 (2006)
6. Laughlin, S.B.: Energy as a constraint on the coding and processing of sensory information. Current Opinion in Neurobiology 11(11), 475–480 (2001)
7. Thiruvarudchelvan, V., Crane, J.W., Bossomaier, T.: Analysis of spikeprop convergence with alternative spike response functions. In: IEEE Symposium on Foundations of Computational Intelligence (April 2013)
8. Esmaeilzadeh, H., Cao, T., Yang, X., Blackburn, S.M., McKinley, K.S.: Looking back and looking forward: power, performance, and upheaval. Commun. ACM 55, 105–114 (2012)
9. Bache, K., Lichman, M.: UCI Machine Learning Repository (2013), `http://archive.ics.uci.edu/ml`
10. Sporea, I., Grüning, A.: Reference time in spikeprop. In: The 2011 International Joint Conference on Neural Networks (IJCNN), July 31-August 5, pp. 1090–1092 (2011)
11. Wilson, D., Martinez, T.R.: The general inefficiency of batch training for gradient descent learning. Neural Networks 16(10), 1429–1451 (2003)
12. Schrauwen, B., Van Campenhout, J.: Extending spikeprop. In: Proceedings of the IEEE International Joint Conference on Neural Networks, vol. 1, p. 4, vol. (xlvii+3302) (July 2004)
13. McKennoch, S., Liu, D., Bushnell, L.: Fast modifications of the spikeprop algorithm. In: International Joint Conference on Neural Networks IJCNN 2006, pp. 3970–3977 (2006)

Robust Fast Online Multivariate Non-parametric Density Estimator

Yoshihiro Nakamura[1] and Osamu Hasegawa[2]

[1] Department of Computational Intelligence and System Science, Tokyo Institute of Technology, Yokohama, Japan
nakamura.y.ba@m.titech.ca.jp
[2] Imaging Science and Engineering Laboratory, Tokyo Institute of Technology, Yokohama, Japan
hasegawa.o.aa@m.titech.ac.jp

Abstract. With the recent development of network and sensor technologies, vast amounts of data are being continuously generated in real time from real-world environments. Such data includes in many noise, and it is not easy to predict that distribution underlying the data in advance. Probability density estimation is a critical task of machine learning, but it is difficult to accomplish it for big data in the real world. For handling such data, we propose a robust fast online multivariate non-parametric density estimator. Our proposed method extends the kernel density estimation and Self-Organizing Incremental Neural Network. The experimental results show that our proposed method outperforms or achieves a state-of-the-art performance.

1 Introduction

With the recent increased development of network and sensor technologies, vast amounts of data are being continuously generated in real time from innumerable real-world environment. Such data are called "big data" and are expected to be analyzed and utilized in many ways.

When analyzing data, a probability density estimator is often used, but it is difficult to use this estimator with conventional methods for big data. Density estimators for big data should satisfy the three conditions described below.

First, density estimators should be fast online learning methods that can sequentially learn samples. The essence of big data is the velocity of data generation. Vast amounts of data are generated very fast, which leads to accumulation of large amount of data. Because these data are generated in real time on a massive scale, they cannot be analyzed using batch learning methods.

Second, density estimators should be non-parametric approaches which do not require an initial distribution underlying observed data to be supplied. The purpose of analyzing big data is primarily for data mining and knowledge discovery. In many cases, it is not easy to predict in advance the distribution underlying the observed data. Therefore, we cannot make use of parametric density estimators which assume that certain known distributions generate observed samples.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 180–187, 2013.

If this assumed distribution does not correspond with the distribution of the observed data, the estimators' performance severely degrades.

Third, density estimators should have high levels of robustness. Data are generated from the real-world environments and often include noise. There are two types of noise: (1) noise that must be realized via probability distributions, such as the variance and fluctuation in a phenomenon; and (2) noise generated from the environments, which have to be eliminated, such as outliers, abnormal samples, and samples generated from diffuse or non abundant distributions. The function to eliminate noise (2) is robustness [1]. To analyze data from the environments, such analysis methods need high robustness because noise included in training data results in over-fitting and a decrease in performance.

There is no conventional method that simultaneously satisfies these three points. The typical non-parametric density estimation technique is kernel density estimation (KDE) [2]. Given training samples $\{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^d,\ i = 1, 2, \cdots, N\}$, the kernel density estimator is

$$\hat{p}(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i), \tag{1}$$

where K is a kernel function. The Gaussian kernel

$$K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{\mu}) = \left((2\pi)^d |\boldsymbol{H}|\right)^{-\frac{1}{2}} \exp\left(-1/2\ (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{H}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right) \tag{2}$$

is often used as K, where $\boldsymbol{H}$ is a parameter called the bandwidth matrix, which materially affects the performance of the estimator. Therefore, numerous optimization approaches exist [3–5] However, because these methods are batch methods, they cannot handle big data. With an increase in the size of the training data, the time and space complexity of KDE increases. Furthermore, to attain high performance, KDE requires many samples [3]. Hence, there is a trade-off between complexity and performance. There are methods that decrease complexity to introduce

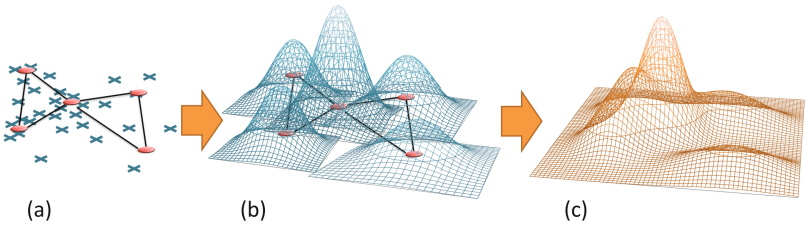


Fig. 1. Outline of proposed method: (a) learn samples online as networks of sample prototypes; (b) determine the shape and size of each kernel on a node by the local structure of a network around the node; and (c) estimate the probability density function as a linear combination of kernels.

online learning [6, 7]. however, their training speed is not fast and they do not have high robustness. equation (1) shows that if training samples include noises, KDE locates kernels in noises, which in turn causes over-fitting and leads to the decrease in performance. Kim et al. [1] consider KDE as one of the kernel methods and optimize it using robust estimation, and realize robust KDE. However, this method is a batch method, and therefore cannot handle big data.

In this study, to handle big data, we propose a non-parametric density estimation approach that accomplishes fast online learning and has high robustness.

2 Proposed Methods

Our proposed method is an extension of KDE and Self-Organizing Incremental Neural Network (SOINN) [8,9]. The outline of our proposed method is illustrated in Fig.1. Our method learns samples as prototype networks and estimate density function using the networks.

2.1 Self-Organizing Incremental Neural Network

SOINN is a online learning method based on network structure of prototypes. It can learns samples from noisy complex non-stationary distributions as a network structure in which the number of nodes is much less than that of samples. It doesn't requires the initialization of the number of nodes and location of nodes; therefore, it can be applied to incremental learning. There are some versions of SOINN. In this paper, Adjusted SOINN [9] is called SOINN.

SOINN have been used for clustering method in many applications, but we consider that SOINN's networks has the information regarding sample distribution and it can be extended to density estimation.

When inputting a sample to SOINN online in real time, its nodes perform competitive learning on the sample. On the basis of this, nodes are inserted, deleted, and moved, and then edges are added and deleted as necessary. Then, SOINN's networks are updated to approximate samples distribution. Each node is behalf of samples which are counted to the node in competitive learning. The node is located on the mean of these samples. An Edge exists between nodes when the frequency of samples between the nodes is relatively high and the samples of which the nodes are representative scatter in the direction of the edge. We perceive each node as a prototype that is representative of samples around the node, and that the network around the node expresses the breadth and direction of spread of the samples.

2.2 Density Estimator

Our proposed method estimates the density function by using information expressed by the SOINN's networks shown above in 2.1.

The density estimator of our proposed method $\hat{p}(\boldsymbol{x})$ is defined as follows :

$$\hat{p}(\boldsymbol{x}) = \frac{1}{T_{\mathcal{N}}} \sum_{n \in \mathcal{N}} t_n K_{\boldsymbol{C}_n}(\boldsymbol{x} - \boldsymbol{w}_n) \tag{3}$$

Table 1. Definitions of variables and symbols

$\boldsymbol{\xi}$	Input sample.	$\psi(t)$	$\psi(t) = 1/t$. A function to decide a coefficient of local optimization.
$\mathcal{N}$	The set of all nodes.		
$\mathcal{E}$	The set of all edges. $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$.	Θ_i	The threshold of node i to create a node and edge.
$\mathcal{P}_i$	The set of nodes connected to node i.	λ	A parameter for deleting nodes.
$\boldsymbol{w}_i$	The positional vector of node i.	age_{max}	A parameter for deleting edges.
t_i	The winning times of node i in competitive learning.	ρ	A parameter for thresholds.
		$\boldsymbol{I}$	Identity matrix.
a_e	The age of edge e.	$\lvert\mathcal{S}\rvert$	The number of elements of set $\mathcal{S}$

Algorithm 1. Learning algorithm

if initialization **then**
  $\mathcal{N} \leftarrow \{c_1, c_2\} : c_1$ and c_2 are samples selected randomly from learning data set.
  $\mathcal{E} \leftarrow \phi$
end if
while There is $\boldsymbol{\xi} \in \mathbb{R}^n$. **do**
  $s_1 \leftarrow \arg\min_{c \in \mathcal{N}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$
  $s_2 \leftarrow \arg\min_{c \in \mathcal{N} \setminus \{s_1\}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$
  Calculate thresholds Θ_{s_1}, Θ_{s_2} with equation (5)
  if $||\boldsymbol{\xi} - \boldsymbol{w}_{s_1}|| > \Theta_{s_1}$ OR $||\boldsymbol{\xi} - \boldsymbol{w}_{s_2}|| > \Theta_{s_2}$ **then**
    if $||\boldsymbol{\xi} - \boldsymbol{w}_{s_1}|| < \frac{\Theta_{s_1}}{4}$ **then**
      $t_{s_1} \leftarrow t_{s_1} + 1$
      $\boldsymbol{w}_{s_1} \leftarrow \boldsymbol{w}_{s_1} + \psi(t_{s_1})(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})$
    else
      $\mathcal{N} \leftarrow \mathcal{N} \cap \{\boldsymbol{\xi}\}$
    end if
  else
    if $(s_1, s_2) \notin \mathcal{E}$ **then**
      $\mathcal{E} \leftarrow \mathcal{E} \cap \{(s_1, s_2)\}$
    end if
    $a_{(s_1,s_2)} \leftarrow 0$
    $a_{(s_1,i)} \leftarrow a_{(s_1,i)} + 1(\forall i \in \mathcal{P}_{s_1})$
    $t_{s_1} \leftarrow t_{s_1} + 1$
    $\boldsymbol{w}_{s_1} \leftarrow \boldsymbol{w}_{s_1} + \psi(t_{s_1})(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})$
    delete edges $\mathcal{E}_{old} = \{e \mid e \in \mathcal{E},\ a_e > age_{max}\}$
    delete nodes $\{i \mid \exists j, (i,j) \in \mathcal{E}_{old}, |\mathcal{P}_i| = 0\}$
  end if
  if the number of input signals is the multiple of λ **then**
    delete nodes $\{i \mid |\mathcal{P}_i| = 0\}$
  end if
end while

where $T_{\mathcal{N}} = \sum_{n \in \mathcal{N}} t_n$ and K is the Gaussian kernel function equation (2). Our proposed method locates kernels on nodes and estimates the density function as a linear summation of kernels. The kernel located on node n is weighted by winning times t_n, where the number of samples represented by node n. The covariance matrix $\boldsymbol{C}_n$ which decides the shape of Gaussian kernel is adaptively determined by the network around node n. We refer to this the "local network covariance matrix," and define it as follows:

$$\boldsymbol{C}_n = \frac{1}{T_{\mathcal{P}_n}} \sum_{p \in \mathcal{P}_n} t_p (\boldsymbol{w}_p - \boldsymbol{w}_n)(\boldsymbol{w}_p - \boldsymbol{w}_n)^T, \tag{4}$$

where $T_{\mathcal{P}_n} = \sum_{i \in \mathcal{P}_n} t_i$ The local network covariance matrix is an extension of the original covariance matrix. It approximates the spread of samples around n.

2.3 Learning Algorithm

Algorithm 1 shows the learning algorithm of our proposed method. One of the extensions of the learning algorithm from SOINN is the decision-making part regarding threshold regions which is arguably the most important part of the SOINN algorithm to add new node and edge.

SOINN threshold region spreads in all directions, including the direction without edges where samples relatively do not exist. This in turn can cause the algorithm to create a long edge unrelated to the density of samples. This edge may have harmful effects on the creation of networks which appropriately express density.

To eliminate network distortion, our proposed method identifies threshold region of a node conforming to the local structure of the network around the node. Θ_i the threshold of node i is decided as follows:

$$\Theta_i = \sqrt{\boldsymbol{d}^T \boldsymbol{M} \boldsymbol{d}}, \tag{5}$$

where

$$\boldsymbol{d} = \frac{\boldsymbol{\xi} - \boldsymbol{w}_i}{||\boldsymbol{\xi} - \boldsymbol{w}_i||}, \quad \boldsymbol{M} = \boldsymbol{C}_i + \rho\gamma\boldsymbol{I}, \quad \gamma = \begin{cases} \min\limits_{p \in \mathcal{N} \setminus \{i\}} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (\mathcal{P}_i = \phi) \\ \frac{1}{|\mathcal{P}_i|} \sum_{p \in \mathcal{P}_i} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (otherwise) \end{cases}.$$

A threshold region of each node is a unit hypersphere transformed by its local network covariance matrix.

3 Experiments

We conducted experiments to compare our proposed method with the state-of-the-art methods in terms of robustness, learning time, and estimate accuracy. All experiments were performed on a PC with 3.20 GHz CPU×8 and 8.00 GB RAM; the implementation language was MATLAB.

3.1 Robustness

We evaluated the accuracy of estimate when methods learn from noisy data. The training data we used were $\boldsymbol{X} \sim f = (1-\alpha)f_0 + \alpha f_1$, where f_0 is a true distribution, and f_1 is a contaminating distribution, with α being the proportion of contamination. Each method learns X and estimates f_0. We evaluated how the accuracy of estimate changed as α increased. We used JS divergence as our evaluation measure. True distribution $f_0(\boldsymbol{x}) = N(\boldsymbol{x}; \boldsymbol{t}, 0.2 \cdot \boldsymbol{I})$, where $\boldsymbol{t} = [a, \sin(3a)]^T, a \in [-2, 2]$. Contaminating distribution f_1 is a uniform distribution whose range of each dimension is $[-4, 4]$. Training data consisted of 5,000 samples, whereas, test data consisted of 20,000 samples. For each value of α, we conducted experiments 20 times and evaluated the mean. We compare our proposed method with other three methods: an online approach of KDE (oKDE) [7], a batch approach of KDE that optimize bandwidth matrix with cross-validation via maximum-likelihood criterion (CVML) [5], a batch approach of KDE that takes robustness into account (RKDE) [1]. Since RKDE requires a bandwidth matrix optimized by other method, we used the bandwidth matrix optimized by CVML for RKDE. Parameters of methods are set as follows: $\lambda = 300, age_{max} = 50, \rho = 0.1, D_{th} = 0.01$.

Fig.2 shows the results of the experiments. When $\alpha = 0$, our proposed method achieved comparable performance to other methods. As α increased, the accuracy of oKDE and CVML decreased, but the accuracy of our proposed method did not, which indicated that our proposed method has a high level of robustness.

3.2 Training Time

We evaluated increases in training time due to increases in the number of samples in the training data. The experimental setup was the same as that in 3.1, but the training data do not contain noise. Finally, the number of samples in the training data was increased from 1,000 to 100,000.

Fig.3 is the experimental results. Fig.3(a) shows training time. Since CVML that is a batch method takes too much time, we stopped the experiment of CVML when the number of training samples was 10,000. Then, our proposed method is approximately 1277 times faster than CVML. To learn 100,000 samples, proposed method is 47.6 times faster than oKDE.

Fig.3(b) shows training time increments. Online approaches incrementally learn samples. We evaluated how much time each method took to learn additional 1,000 samples. training time increment of oKDE increases with the numbers of samples learned thus far. Compared to this, increases in proposed method are minimal, indicating that our proposed method has a high level of scalability.

3.3 Density Estimation of Real Data

We compared our proposed method with other methods used above in 3.1 in terms of density estimation with real data-sets obtained from the UCI Machine Learning Repository. We used data-sets named Iris, Pima, Wine, Wine_Red,

and Wine_White. We conducted each experiment with samples from each class of each data set; furthermore, samples were normalized by whitening. Next, 75% samples were randomly selected to be used to training data, whereas the rest were used as test data. Each experiment was conducted 20 times, and we evaluated the mean of the results of the data sets. The evaluation measure is negative log likelihood, and the parameters of proposed method and oKDE are set as follows: $\lambda = 300, age_{max} = 50, \rho = 0.5, D_{th} = 0.1$.

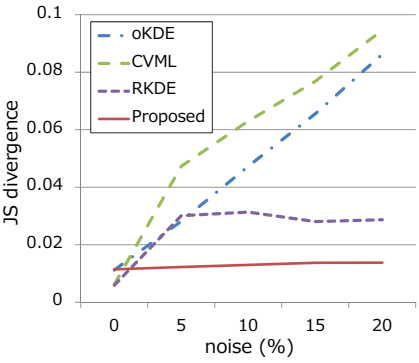
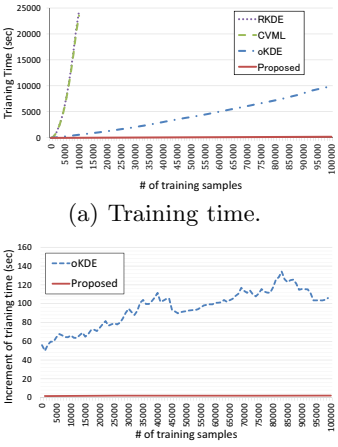


Fig. 2. The experimental results evaluating robustness. We compared our proposed method with oKDE [7], CVML [5], and RKDE [1]. The horizontal axis indicates α, the ratio of noise in training samples.



(a) Training time.

(b) Increment of training time.

Fig. 3. The experimental result about increases in training time

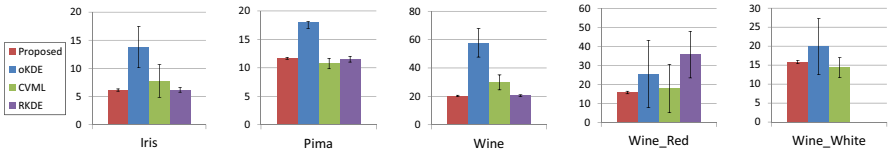


Fig. 4. The experimental results of estimating density functions of real data. Evaluation measure is negative log likelihood (NLL), and bar charts show the means of NLL; error bars show the standard deviations of NLL.

Fig.4 shows the results of the experiments. The performance of our proposed method with some data sets is superior to that of the other method, whereas with some data sets, other methods outperform our proposed method. The performance of our proposed method is comparable when standard deviations are considered. The standard deviations of our proposed method are smaller than those of the other methods, indicating that our proposed method was more stable than the other methods. These results show that our proposed method can be useful in estimating density for real data.

4 Conclusion

In this study, to handle big data, we proposed a robust fast online non-parametric density estimation technique. Our proposed method decreases time and space complexities by creating networks of sample prototypes along sample distributions. The experimental results show that our proposed method considerably outperforms the state-of-the-art methods in terms of robustness and learning times. Furthermore, our proposed method achieves a performance comparable to that of other approaches in terms of estimate accuracy. Because it is difficult to successfully handle noise in real time given huge amounts of data, analysis methods that have high level of robustness and that can handle high-speed data are necessary. In this respect, we feel that our proposed method has a considerably broader range of applications.

Acknowledgement. This work was sponsored by Japan Science and Technology Agencys CREST.

References

1. Kim, J., Scott, C.D.: Robust kernel density estimation. Journal of Machine Learning Research 13, 2529–2565 (2011)
2. Parzen, E.: On estimation of a probability density function and mode. The Annals of Mathematical Statistics 33, 1065–1076 (1962)
3. Silverman, B.: Density Estimation. Chapman and Hall (1986)
4. Jones, M.C., Marron, J.S., Sheather, S.J.: A brief survey of bandwidth selection for density estimation. Journal of the American Statistical Association 91(433), 401–407 (1996)
5. Murillo, J., Rodriguez, A.A.: Algorithms for gaussian bandwidth selection in kernel density estimators. In: Neural Information Processing Systems (2008)
6. Tabata, K., Sato, M., Kudo, M.: Data compression by volume prototypes for streaming data. Pattern Recognition 43, 3162–3176 (2010)
7. Kristan, M., Leonardis, A., Skočaj, D.: Multivariate online kernel density estimation with gaussian kernels. Pattern Recognition 44, 2630–2642 (2011)
8. Shen, F., Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning. Neural Networks 19(1), 90–106 (2006)
9. Shen, F., Hasegawa, O.: A fast nearest neighbor classifier based on self-organizing incremental neural network. Neural Networks 21(10), 1537–1547 (2008)

Digital Dynamical Systems of Spike-Trains

Narutoshi Horimoto and Toshimichi Saito

Hosei University, Koganei, Tokyo, 184-8584 Japan
narutoshi.horimoto.8c@stu.hosei.ac.jp

Abstract. This paper studies a simple digital dynamical system that can generate various spike-trains. In order to consider the steady and transient states, we use two basic feature quantities. The first one is the number of co-existing periodic spike-trains that can characterize richness of the steady state. The second one is the concentricity of transition to the periodic spike-trains that can characterize variation of transient phenomena. Performing numerical experiments for two typical examples based on the bifurcating neuron, basic classification of the dynamics is considered.

Keywords: spiking neurons, digital dynamical systems, spike-train stability, nonlinear dynamics.

1 Introduction

This paper studies the digital spike-phase map (DPM) that is a simple digital dynamical system from a set of one-dimensional lattice points to itself [1] [2]. Depending on parameters and initial condition, the DPM can generate a variety of periodic spike-trains (PSTs) and transient spike-trains (TSTs) to the PSTs. Motivations for studying the DPM are many, including the following. First, the DPM can be a simple example that can contribute to develop study of digital dynamical systems [3]-[5]. The DPM is a digital version of the analog maps that have been contributed to develop study of analog dynamical systems [6]. Second, the spiking signals play important roles in various systems. Analysis of the spike-trains is basic for considering spike-based information processing functions and for developing spike-based engineering applications, e.g., signal processing, spike-based communication, modeling and neural prosthesis [7]-[12]. Third, the DPM has advantages such as robust hardware implementation by the digital circuits and precise computer-aided design/analysis of the circuits. The DPM can have a huge variety of configuration and can output various spike-trains. The results of the analysis can be basic information for modeling various spiking neuron dynamics and for design of reconfigurable hardware of spike-train [8] [9] [13].

In order to consider the spike-trains, we use two basic feature quantities. The first quantity is the number of co-existing PSTs (#PST). The DPM can have plural PSTs and exhibits either of them depending on the initial condition. The #PST can characterize richness of the PSTs. The second quantity is the concentricity of transition to the PSTs (C_t). This quantity can characterize variety

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 188–195, 2013.

of the transient phenomena. This quantity is based on the concentricity of state transition in random neural networks [14]. We then analyze DPMs based on analog spike-phase map (APMs) of the bifurcating neuron (BN). The BN is a simple spiking neuron model that can output various periodic/chaotic analog spike trains. Especially, we consider two kinds of DPMs: the first DPM is based on smooth APM and the second DPM is based on a piecewise linear (PWL) APM. Performing numerical experiments for the DPMs, we give basic classification of the spike-train dynamics. As a typical example, the first DPM (respectively, the second DPM) has a large number of different PSTs (respectively, similar PSTs) in the case where the smooth APM (respectively PWL APM) exhibits chaotic spike-train. Note that our previous papers do not discuss the second quantity and comparison of the two kinds of DPMs.

2 Digital Spike-Phase Maps

The digital spike-phase map (DPM) is defined by

$$\theta_{n+1} = f(\theta_n),\ f : L_1 \to L_1,\ L_1 \equiv \{l_1, l_2, \cdots l_N\},\ l_i \equiv \frac{i-1}{N},\ i = 1 \sim N \quad (1)$$

where f is a mapping from a set of lattice point L_1 and θ_n is the n-th digital spike-phase in the normalized period 1. Iteration of f generates the sequence of the phases $\{\theta_n\}$. Since each of N lattice points has one image, the DPM has N^N variations and can exhibit various phenomena. Using the phases, we define the digital spike-train

$$Y(\tau) = \begin{cases} 1 & \text{for } \tau = \tau_n \\ 0 & \text{for } \tau \neq \tau_n \end{cases} \quad \tau_n = \theta_n + n - 1 \in \{l_1 + n - 1, \cdots, l_N + n - 1\} \equiv L_n \quad (2)$$

For the DPM, we give basic definitions of the steady state. A point $p \in L_1$ is said to be a periodic point (PEP) with period k if $p = f^k(p)$ and $f(p)$ to $f^k(p)$ are all different where f^k is the k-fold composition of f. The PEP with period 1 is referred to as a fixed point. A sequence of the PEPs $\{p, f(p), \cdots, f^{k-1}(p)\}$ is said to be a periodic orbit (PEO) with period k. Note that a PEP with period k corresponds to one PST with period k and that a PEO with period k corresponds to k PSTs. Figure 1 shows an example of the DPM. As shown in the figure, the one fixed point corresponds to one PST, the one PEO with period 2 corresponds to 2 PST and one PEO with period 3 corresponds to 3 PSTs. This correspondence is different from analog 1D map whose one PEO corresponds to one steady state [6]. In general, one PEO with period k consists of k PEPs, $\{f(p), \cdots, f^k(p)\}$. Each PEP can be an initial spike position in L_1 that gives one PST with period k $\{\tau_1, \cdots, \tau_k\}$ such that

$$\tau_1 = f^m(p) \in L_1, \tau_2 = f^m(p) + 1 \in L_2,\ \cdots,\ \tau_k = f^{m+k}(p) + (k-1) \in L_k \quad (3)$$

where $m = 1 \sim k$ and $\tau_{k+1} = \tau_1 + k$. Each of these PSTs is coincident with other ones by phase shift.

Next, we give basic definitions of the transient state. A point $q \in L_1$ is said to be an eventually periodic point (EPP) with step k if the q is not a PEP but falls into some PEP p after k steps: $f^k(q) = p$. An EPP corresponds to an initial spike-position that gives a transient spike-train to the PST. The EPP represents a TST and characterizes the domain of attraction to the PST. Figure 1 shows an EPP that fall into the PEP with period 2. Corresponding TST is also shown.

3 Basic Feature Quantities

In order to analyze dynamics of the DPM and PST, we introduce two basic feature quantities. The first quantity is a basic one: the number of PSTs (#PST) that can characterize richness of the steady state. #PST is given by the number of PEPs of the DPM.

$$\#\mathrm{PST} = \#\{\text{PEPs of } f\} \tag{4}$$

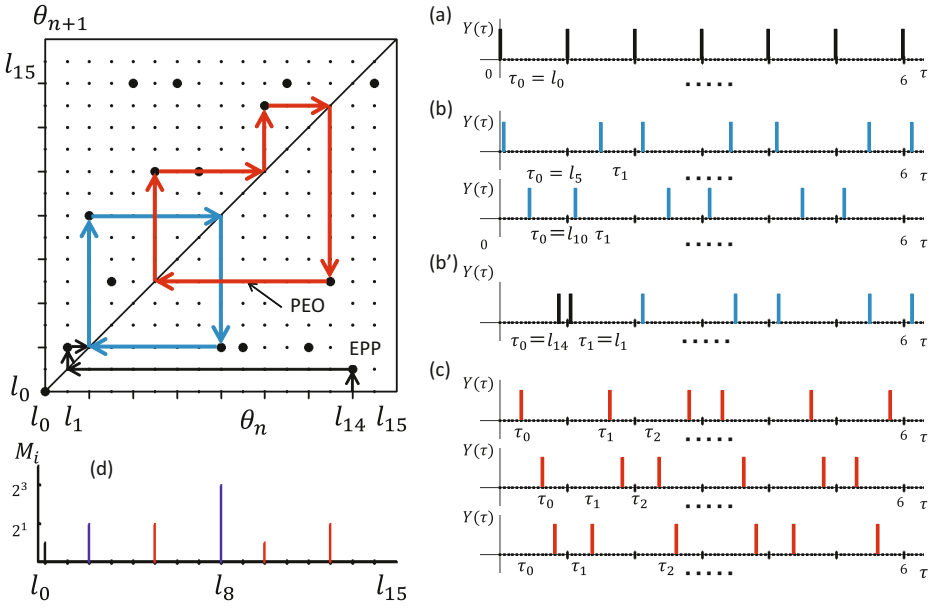


Fig. 1. Digital spike-phase map (DPM). The fixed point (black), the blue PEO with period two, and the red PEO with period three correspond to one PST in (a), two PSTs in (b), and three PSTs in (c), respectively. Eventually periodic point (l_{14}) corresponds to the TST in (b'). (d) Distribution of EPPs. #PST=6, $C_t = 30/6 = 5$. The 1st PEP is l_0 with 1 EPP ($M_1 = 1$), the 2nd PEP is l_2 with 2 EPPs ($M_2 = 2$), the 3rd PEP is l_5 with 2 EPPs ($M_3 = 2$), the 4th PEP is l_8 with 8 EPPs ($M_4 = 8$), the 5th PEP is l_{10} with 1 EPP ($M_5 = 1$) and the 6th PEP is l_{13} with 2 EPPs ($M_5 = 2$),

The DPM can have plural PSTs and exhibits one of them depending on the initial condition. Figure 1 illustrates an example of the DPM where #PST=6.

The second quantity is the concentricity of EPPs that can characterize the variation of the domain of attraction. In the case where the DPM has N_p PEPs corresponding to N_p PSTs, let M_i denote the number of EPPs that fall into the i-th PEP where $i = 1 \sim N_p$. Figure 1 illustrates distribution of the EPPs for $N_p = 6$. The second quantity is the 2nd moment of the distribution of the EPPs:

$$C_t = \frac{1}{N_p} \sum_{i=1}^{N_p} M_i^2 \tag{5}$$

Figure 1 (d) illustrates a distribution of EPPs. This DPM gives $C_t = 30/5$. We refer to C_t as the concentricity of spike-position transition. This quantity is based on the concentricity of state transition in random neural networks [14].

4 Examples Based on the Bifurcating Neurons

We introduce a typical example of the DPM based on the BN that a simple spiking neuron model. Repeating integrate-and-fire behavior between the constant threshold and periodic base signal $b(\tau)$ as shown in Fig. 2, the BN can output various chaotic/periodic spike-trains [15] [16]. Let τ_n denote the n-th spike-position of the BN with base signal $b(\tau)$ with period 1. Let $\theta_n = \tau_n \bmod 1$ be the n-th spike-phase. The spike-phase is governed by the analog spike-phase map (APM)

$$\theta_{n+1} = \theta_n + 1 - b(\tau_n) \bmod 1 \equiv g_a(\theta_n) \tag{6}$$

where $\theta_n \in [0, 1)$ and $b(\tau) = b(\tau + 1)$. $|b(\tau)| < 1$ is satisfied for all τ. Note that the BN can output various spike-trains depending on the shape of $b(\tau)$. This APM is a typical example of nonlinear dynamical systems that can exhibit various peiorid/chaotic phenomena. For simplicity, we consider two examples: a sinusoidal base signal $b_1(\tau) = -k \sin 2\pi\tau$ and a PWL triangular base signal

$$b_2(\tau) = \begin{cases} -4k\tau & \text{for } -\frac{1}{4} \le \tau < \frac{1}{4} \\ 4k(\tau - \frac{1}{2}) & \text{for } \frac{1}{4} \le \tau < \frac{3}{4} \end{cases} \quad \text{for } 0 \le \tau < 1, \quad b_2(\tau) = b_2(\tau + 1) \tag{7}$$

where $0 < k < 1$. Figure 2 shows dynamics for these base signals. Figure 3 shows examples of the APMs. For the sinusoidal base signal $b_1(\tau)$, the APM is smooth. As k increases, the stable fixed point is changed into chaotic orbit via period doubling bifurcation. For the triangular base signal $b_2(\tau)$, the APM is piecewise linear. As k increases, the stable fixed point is changed suddenly into chaotic orbit. The stable fixed point corresponds to a stable periodic spike-train with period 1. In the same manner as the APMs, the stable periodic spike-train of the BN is changed into chaotic/periodic spike-trains.

Discretizing the APM, we obtain the DPM:

$$\theta_{n+1} = \frac{1}{N} \mathrm{INT}(N g_a(\theta_n) + 0.5) \equiv g_d(\theta_n) \tag{8}$$

where $\theta_n \in L_0$ and INT(X) means the integer part of X. This is a concrete example of the DPM defined in Eq. (1). As N varies, the dynamics of the DPM varies in extremely complex ways. For simplicity, this paper studies the case $N = 128$. Figure 5 (a) shows the DPM for $N = 128$ corresponding to the APM of sinusoidal base signal in Fig. 4 (b). This smooth APM exhibits chaotic orbit whereas the corresponding DPM has 22 PEPs (22 PSTs). Figure 5 (b) shows the DPM for $N = 128$ corresponding to the APM of triangular base signal in Fig. 4 (d). This PWL APM exhibits chaotic orbit whereas the corresponding DPM has 42 PEPs (42 PSTs). In general, as compared with the APMs, the DPMs tend to have larger number of steady state (PSTs) and exhibit either one depending on the initial state.

5 Basic Classification

We try to give basic classification of the DPMs using the feature quantities: the number of PSTs (#PST = #PEP) and concentricity of transition to the PSTs (C_t). Although detailed classification is hard, this paper shows two cases: simple dynamics and complex dynamics.

Case 1: #PST is small and C_t is very large. In this case, almost all initial states fall into a small number of PST (PSP) and the TSTs concentrate into small number of PSTs. A long transient phenomenon can exist. Figure 4 shows typical examples. In the figure, orbits started from almost all EPP fall into one fixed point. In the sinusoidal base signal $b_1(\tau)$ for $k = 0.159$ and the triangular base signal $b_2(\tau)$ for $k = 0.375$, the APM has one stable fixed point (Fig. 3 (a) and (c)) whereas DPM has two fixed points (Fig. 4 (a) and (b)). In the DPM, orbit from all the EPPs fall into the one fixed point at lattice l_{63}. The distribution of the number of EPPs (M_i) is almost δ function and C_t is very large. The corresponding steady state PST consists of equidistant inter-spike intervals with period 1. In these examples, dynamics based on the sinusoidal and triangular bases are the almost same.

Case 2: #PST is large and C_t is (very) small. In this case, the DPM has a large number of PSTs and exhibits either PST depending on the initial states. That is, the DPM can have rich PSTs. The distribution of the number of EPPs (M_i) does not concentrate into a small number of PEPs (PSTs). The domains of attraction are not wide, however, some transients can be long. Such a situation may be suitable for spike-based coding with error correction function. Figure 5 shows typical examples in this case. In the sinusoidal base signal $b_1(\tau)$ for $k = 0.705$, the APM exhibits chaotic orbit (Fig. 3 (b)) whereas the DPM has 22 PSTs (Fig. 5 (a)): two PSTs with period 4, two PSTs with period 3, three PSTs with period 2 and two PSTs with period 1 (fixed points). In the triangular base signal $b_2(\tau)$ for $k = 0.705$, the APM has chaotic orbit (Fig. 3 (d)) whereas DPM has 44 PSTs (Fig. 5 (b)): 20 PSTs with period 20 and 2 PSTs with period 1 (fixed points). The distribution of the number of EPPs (M_i) is shown in Fig. 5 and C_t is small. Note that the sinusoidal-based DPM and triangular-based

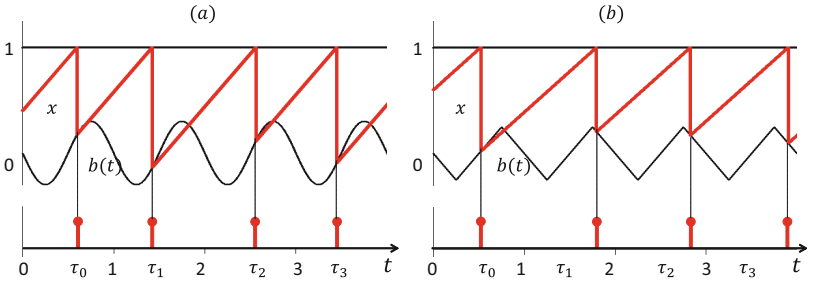


Fig. 2. Integrate-and-fire dynamics of the bifurcating neuron (BN) with (a) sinusoidal base signal $b_1(\tau)$ and (b) triangular base signal $b_2(\tau)$

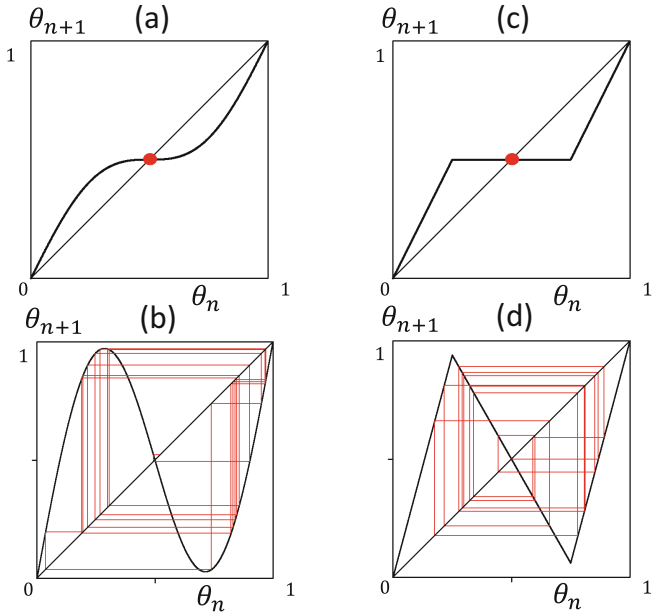


Fig. 3. Examples of analog phase maps (APMs). sinusoidal base signal $b_1(\tau)$: (a) $k = 0.159$, (b) $k = 0.705$. Triangular base signal $b_2(\tau)$: (c) $k = 0.375$, (d) $k = 0.705$.

DPM have differences. First, except for the fixed points, the number of PEO is 6 and 2. The triangular-based DPM has large number of PSTs, however, they are coincident to each other by phase shift. Second, the distribution of M_i of the sinusoidal-based DPM has larger variation (a variety of domains of attraction) than the triangular-based DPM. Note that the DPMs can exhibit a huge variety of dynamics (#PST and C_t vary in a very complex way) as k varies. However, the analysis is very hard and will be considered in the future.

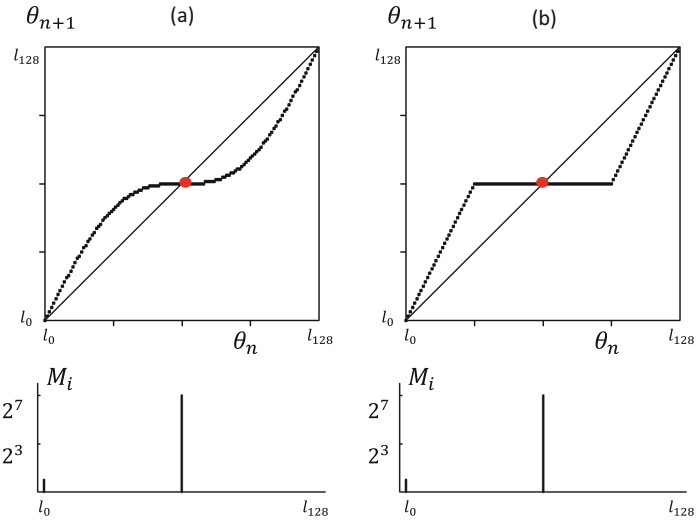


Fig. 4. Typical DPMs corresponding to APMs with stable fixed point. (a) $k = 0.159$, #PEP=#PST=2, $C_t = 8065$ (b) $k = 0.375$, #PEP=#PST=2, $C_t = 8065$

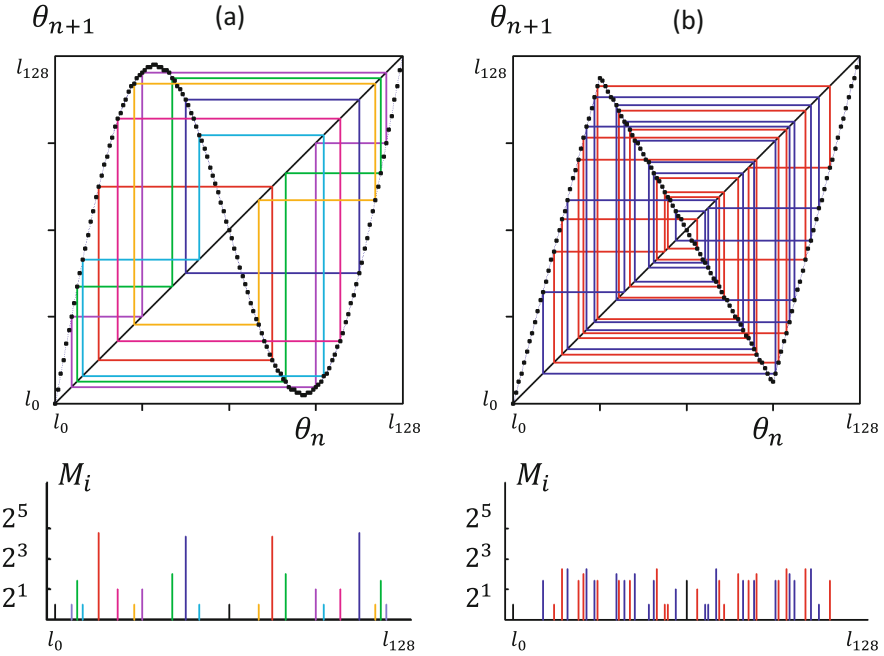


Fig. 5. Typical DPMs corresponding chaotic APMs. The PEOs are distinguished by their colors. (a) $k = 0.705$, #PEP=#PST=22, $C_t = 108.9$ (b) $k = 0.705$, #PEP=#PST=42, $C_t = 11.1$

6 Conclusions

Typical dynamics of the DPM has been studied in this paper. Using the two feature quantities, #PST and C_t, two examples of the DPM have been analyzed and basic classification of the dynamics has been given. Especially, in the case where the APM exhibits chaotic spike-train, the corresponding DPM can exhibit a variety of PSTs.

Future problems include analysis of bifurcation phenomena of the DPMs, classification of PSTs with coding capabilities and hardware implementation for engineering applications.

References

1. Horimoto, N., Saito, T.: Analysis of Digital Spike Maps based on Bifurcating Neurons. NOLTA, IEICE, E95-N 10, 596–605 (2012)
2. Ogawa, T., Saito, T.: Self-organizing Digital Spike Interval Maps. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part II. LNCS, vol. 7063, pp. 612–617. Springer, Heidelberg (2011)
3. Chua, L. O.: A nonlinear dynamics perspective of Wolfram's new kind of science, I, II. World Scientific (2005)
4. Wada, W., Kuroiwa, J., Nara, S.: Completely reproducible description of digital sound data with cellular automata. Physics Letters A 306, 110-115 (2002)
5. Ito, R., Nakayama, Y., Saito, T.: Learning of Dynamic BNN toward Storing-and-Stabilizing Periodic Patterns. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part II. LNCS, vol. 7063, pp. 606–611. Springer, Heidelberg (2011)
6. Ott, E.: Chaos in dynamical systems. Cambridge (1993)
7. Campbell, S.R., Wang, D., Jayaprakash, C.: Synchrony and desynchrony in integrate-and-fire oscillators. Neural Computation 11, 1595–1619 (1999)
8. Izhikevich, E.M.: Simple Model of Spiking Neurons. IEEE Trans. Neural Networks 14(6), 1569–1572 (2003)
9. Torikai, H., Funew, A., Saito, T.: Digital spiking neuron and its learning for approximation of various spike-trains. Neural Networks 21, 140–149 (2008)
10. Torikai, H., Nishigami, T.: An artificial chaotic spiking neuron inspired by spiral ganglion cell: parallel spike encoding, theoretical analysis, and electronic circuit implementation. Neural Networks 22, 664–673 (2009)
11. Rulkov, N.F., Sushchik, M.M., Tsimring, L.S., Volkovskii, A.R.: Digital communication using chaotic-pulse-position modulation. IEEE Trans. Circuits Systs., I 48(12), 1436–1444 (2001)
12. Iguchi, T., Hirata, A., Torikai, A.,, H.: Theoretical and heuristic synthesis of digital spiking neurons for spike-pattern-division multiplexing. IEICE Trans. Fundamentals, E93-A 8, 1486–1496 (2010)
13. Matsubara, T., Torikai, H.: Asynchronous cellular automaton-based neuron: theoretical analysis and on-FPGA learning. IEEE Trans. Neiral Netw. Learning Systs. 24, 736–748 (2013)
14. Amari, S.: A Method of Statistical Neurodynamics. Kybernetik 14, 201–215 (1974)
15. Perez, R., Glass, L.: Bistability, period doubling bifurcations and chaos in a periodically forced oscillator. Phys. Lett., 90A 9, 441–443 (1982)
16. Torikai, H., Saito, T., Schwarz, W.: Synchronization via multiplex pulse-train. IEEE Trans. Circuits Syst. I 46(9), 1072–1085 (1999)

Supervised Multiple Timescale Recurrent Neuron Network Model for Human Action Classification

Zhibin Yu, Rammohan Mallipeddi, and Minho Lee*

School of Electronics Engineering, Kyungpook National University
1370 Sankyuk-Dong, Puk-Gu, Taegu 702-701, South Korea
{ahriman1985abr,mallipeddi.ram,mholee}@gmail.com

Abstract. Multiple time-scales recurrent neural network (MTRNN) model is a useful tool to record and regenerate a continuous signal for a dynamic task. However, the MTRNN itself cannot classify different motions because there are no output nodes for classification tasks. Therefore, in this paper, we propose a novel supervised model called supervised multiple time-scales recurrent neural network (SMTRNN) to handle the classification issue. The proposed SMTRNN can label different kinds of signals without setting the initial states. SMTRNN provided both prediction and classification signals simultaneously during testing. In addition, the experiment results show that SMTRNN successfully classifies a continuous signal including multiple kinds of actions as well predicts motions.

Keywords: Human action, SMTRNN, continuous time-scales recurrent neuron network, classification.

1 Introduction

Human action classification plays an important role in human-computer interaction (HCI). Humans can easily recognize the activities of other humans but it is still a challenging task for artificial intelligent systems to do so via mathematical and computational algorithms even with advanced sensor systems. In literature, vision-based recognition has been proved to be an efficient method for human motion classification [9]. Hidden Markov Model (HMM) is considered to be one of the traditional dynamic models to recognize motion sequences [8, 12].

On the other hand, some researchers [10, 13] try to use the "behavior compositionality" [11] to recognize various kinds of actions based on the trajectories sequences. Based on this idea, Yamashita and Tani [14] proposed a neural-dynamic model referred to as multiple time scale recurrent neural networks (MTRNN) to imitate the human motor control system. The MTRNN is an extension of continuous timescale recurrent neural networks (CTRNN) [4] and can predict a continuous signal. The MTRNN possesses an important feature called "self-organization". It is a phenomenon in which a global coherent structure appears in a system not by a central authority but by local interactions among elements of the system [5]. Further research on

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 196–203, 2013.

MTRNN [1, 7] validates its ability to generate some untrained continuous signal based on the existing knowledge.

MTRNN is also applied in signal classification. In [15], the authors use a conventional neural network to analyze the values of slow context units in MTRNN. In addition, they try to use the prediction error to classify known and unknown sound signals. Even though, the experimental results show that different signals are classifiable, they concede that their method based on a 3-layer neural network is inefficient to extract information from slow context units.

By assigning different initial states, MTRNN can easily estimate multiple kinds of signals. However, the estimation performance of MTRNN depends on the accuracy of the initial states corresponding to the signal in both training and testing. The problem of setting the accurate initial states makes MTRNN impossible to classify different motion sequences. In [6], some researchers proposed a recurrent neural network (RNN) model for dynamic signal classification. RNN consists of two kinds of output neurons: prediction neurons and classification neurons. Both work together to output the prediction and classification signals synchronously. This model shows a possible way to classify different dynamic signals.

In this paper, we proposed a new model called supervised multiple time-scales recurrent neural network (SMTRNN) by composing MTRNN and supervised training algorithm. In our proposed model, the initial states of all signals are the same. Instead of the initial states, we appoint a certain area located in slow context layer for output classification using supervised methods. In the current work, we redefine the error function of MTRNN by including the label information and use back propagation though time (BPTT) for training. Unlike most classification methods, the classification output of our model is a continuous sequence where the length of the classification output is equal to the input signal. SMTRNN inherits the "self-organization" capability of MTRNN and is able to classify a long untrained signal including several short trained signal sequences.

Fig. 1 shows the motivation of the proposed model. During training, certain labels are given to SMTRNN along with the training signals which are then used as the target outputs for classification in the slow context layer. In testing stage, in addition to classification, SMTRNN is able to predict and follow human subjects' action at the same time.

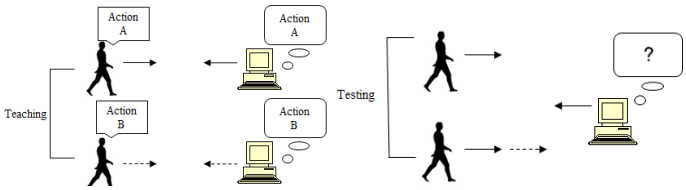


Fig. 1. Motivation of the proposed model

2 Related Works

In MTRNN literature, self-organizing map (SOM) is used as an input/output layer to accept vision signals and output the prediction signals. SOM algorithm has the ability

to preserve topological properties of the input space. In the present work, since the focus is on action recognition, we simplified the model, proposed by Yamashita et al. [7], and employ an 8*8 SOM layer to accept the 40 dimensional visual inputs. In our model, the output of SOM can be regarded as the prediction of human actions.

Context layers, the main components of MTRNN, are modeled by CTRNN. CTRNN is a special type of RNN and is a dynamical system model of biological neural networks. In CTRNN, the output of each neuron is calculated using both the current input samples and the past history of the neural states. Hence, making CTRNN suitable for predicting the continuous sensori-motor sequences [3].

In the current work, Back Propagation Through Time (BPTT) is used for training where the error function is defined using Kullback–Leibler divergence:

$$\mathrm{E} = \sum_t \sum_{i \epsilon O} y^*_{i,t} \log \frac{y^*_{i,t}}{y_{i,t}} \tag{1}$$

where O units in input-output layer, $y^*_{i,t}$ is the desired output value of the i-th neuron at time step t and $y_{i,t}$ is the prediction value of the i-th neuron with the existing weights and the initial states. In our model, we combine the proprioceptive and the vision input into one SOM layer.

The weight updating rule is described by the following equation:

$$w_{ij}(n+1) = w_{ij}(n) - \alpha \frac{\partial E}{\partial w_{ij}} \tag{2}$$

where n is the iteration step, $\alpha$is the learning rate and is set to 0.0005 in our experiments. The partial differential $\frac{\partial E}{\partial w_{ij}}$ is given by:

$$\frac{\partial E}{\partial w} = \sum_t \frac{1}{\tau_i} \frac{\partial E}{\partial u_{i,t}} y_{j,t-1} \tag{3}$$

$$\frac{\partial E}{\partial u_{i,t}} = \begin{cases} y_{i,t} - y^*_{i,t} + \left(1 - \frac{1}{\tau_i}\right) \frac{\partial E}{\partial u_{i,t+1}} & i \in O \\ \sum_{k \in N} \frac{\partial E}{\partial u_{k,t+1}} \left[\delta_{ik} \left(1 - \frac{1}{\tau_i}\right) + \frac{1}{\tau_k} w_{ki} f'(u_{i,t})\right] & otherwise \end{cases} \tag{4}$$

where $f'(x)$ is the derivative of the sigmoid function and δ_{ik} is Kronecker delta($\delta_{ik} = 1$ if $i = k$and otherwise 0).

3 Proposed Model

3.1 Motivation

Before explaining the proposed model, we would like to reveal the possibility of MTRNN based classification. The experiment results presented by the authors in [15] point out that the slow context units can be used for classification. In other words, some of the slow context nodes contain valuable information for classification. However, the identification of the slow context nodes that contain valuable information has a significant problem. In addition, we are not sure if the slow context nodes that

contain valuable information and needed for classification are distributed randomly over the entire slow context layer. To understand the phenomena, we conducted an experiment.

In the experiment, we used MTRNN to predict and classify 3 kinds of actions (running, walking and swinging arms) based on the KINECT [2]. The stimulus output values of slow context nodes are recorded as shown in Fig. 2 (a) ~ (c). The difference between nodes 28 ~30 and nodes 25 ~27 is that the latter ones are assigned to different initial conditions. The initial states for nodes 28, 29 and 30 are set by 100, 010 and 001, respectively, while we use the same initial values for all other nodes. The initial values are only used for training process. In other words, all the initial states (including node 28 ~ 30) of 3 different actions are the same during testing. As shown in Figs. 2 (a), (b) and (c), it seems that 3 different kinds of single actions in the slow context units are classifiable. However, it is hard to recognize the combination of each motion based on conventional MTRNN. Since the MTRNN stores dynamic characteristics with large inertia energy of the first action and it is difficult to quickly adapt a continuous motion change with different dynamic characteristics, which intrinsically induces the unpredictable time delay, the monitoring of initial values of the conventional MTRNN is not enough for classifying the motion change and/or the combination of single motion as shown Fig. 2 (d).

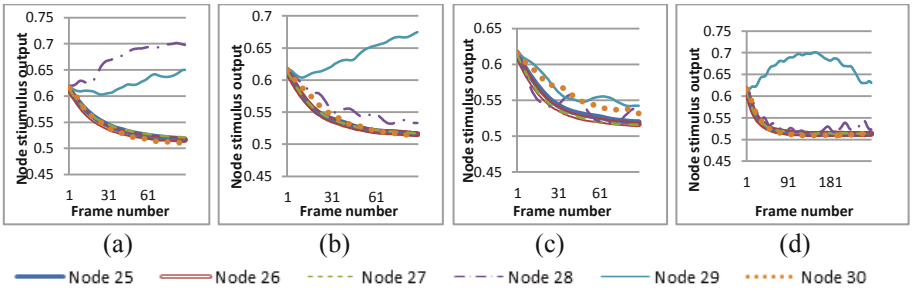


Fig. 2. The nodes related with initial states (node 28 ~30) have different behaviors comparing with other nodes. (a) Running (b) Walking (c) Swinging arms (d) Walking +Swinging

In Fig. 2(d), we analyzed a long continuous signal including two different actions (first walking, then swinging), and the stimulus output was similar with walking condition as Fig. 2 (b). When the human test changed his action to swinging after about 160th frame, the stimulus output of node 29 decreased very slowly. By comparing Fig. 2 (c) with Fig. 2 (d), it's hard to say that we find a swinging motion after 160th frame of Fig. 2 (d). Thus, we wanted to develop a model to overcome these short come sings of previous models. Therefore, in our model we add some constraints which will make some certain nodes to generate stable signals which can be directly assigned to action labels.

3.2 The Proposed Supervised MTRNN

In the proposed model, the classification nodes are a part of slow context and have the same time constant τ. The classification part works similar to the input-output layer.

The purpose of the input-output layer is to accept the current motion signal and output the prediction signal. Classification nodes will accept the label value of last frame and output the label for the current frame. The difference between classification nodes and other slow context nodes is that classification nodes need to back propagate classification error to other nodes. The number of classification nodes is decided by the class number. In our experiments we used 1-N encoding, which is a kind of encoding algorithm used in [6]. Thus the number of classification nodes is equal to the class number. The node with highest output dominates the class label. Considering the definition of error function in MTRNN, we define the error function of SMTRNN as:

$$\mathrm{E} = \sum_t \sum_{i \epsilon O,C} y^*_{i,t} \log \frac{y^*_{i,t}}{y_{i,t}} \tag{5}$$

where O and C denote the prediction and classification output nodes, respectively. Then we can deduce the partial differentialequations as:

$$\frac{\partial E}{\partial u_{i,t}} = \begin{cases} y_{i,t} - y^*_{i,t} + \left(1 - \frac{1}{\tau_i}\right) \frac{\partial E}{\partial u_{i,t+1}} \; i \in O \\ y_{i,t} - y^*_{i,t} + \sum_{k \in N} \frac{\partial E}{\partial u_{k,t+1}} \left[\delta_{ik} \left(1 - \frac{1}{\tau_i}\right) + \frac{1}{\tau_k} w_{ki} f'(u_{i,t})\right] i \in C \\ \sum_{k \in N} \frac{\partial E}{\partial u_{k,t+1}} \left[\delta_{ik} \left(1 - \frac{1}{\tau_i}\right) + \frac{1}{\tau_k} w_{ki} f'(u_{i,t})\right] otherwise \end{cases} \tag{6}$$

The training process of the proposed model is shown in Fig. 3. The classification nodes and input-output nodes work synchronously. The action prediction output of input-output layer can be viewed as a by-product of action classification.

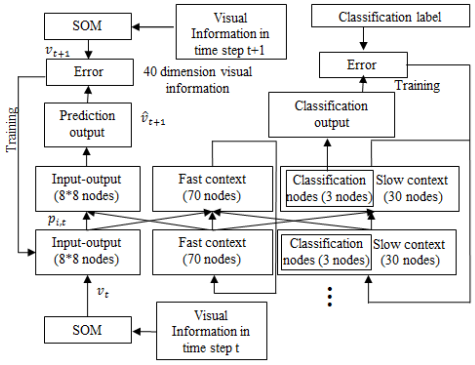


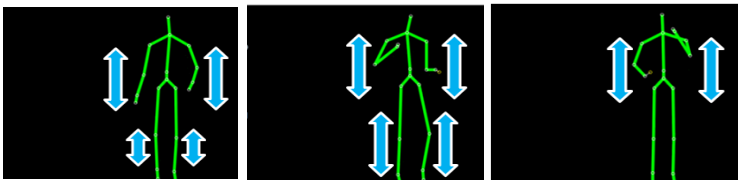
Fig. 3. Training process of the hybrid system

4 Results

Experimental results reported in this paper were performed on an IBM computer with Microsoft Windows 7 running on an Intel Core (TM) 29400 2.67 GHz, 4GB memory. The data corresponding to 3 different human actions: walking, running and swing arms were collected. 10 samples corresponding to each action were collected. Each

action was captured over a time period of 3 seconds (90 frames) using KINECT. The skeletal view of 3 different actions is shown in Fig. 4. In Fig. 4, the blue arrows express the vibration magnitude. As shown in Fig. 4, the vibration magnitude legs during running (Fig. 4 (b)) is relatively larger compared to the other two actions, walking (Fig. 4 (a)) and swinging arms (Fig. 4 (c)).

During the training process, the mean square error (MSE), which is the average square error per neuron per step over all teaching sequences, was reduced. Training was done over 10,000 iterations and the MSE converged to 0.001342. The total training time was about 3 hours.



(a) (b) (c)

Fig. 4. 3 different kinds of human actions obtained by KINECT. (a) corresponds to walking action; (b) corresponds to running action and (c) corresponds to swinging arms action

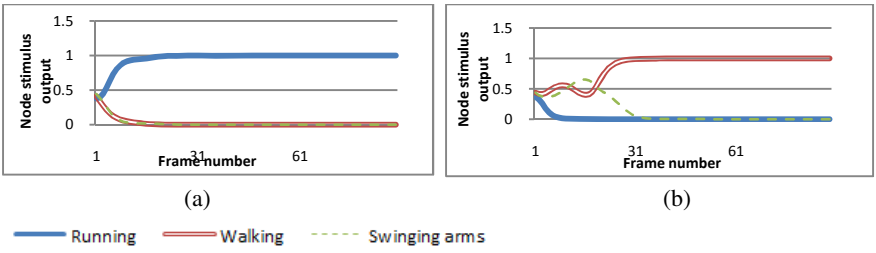


Fig. 5. Two typical classification results

Fig. 5 shows the experimental results. As shown in Fig. 5 (a), due to the time scale feature of MTRNN, the output curve changes gradually and becomes stable after the 13th frame. In Fig. 5 (a), the proposed model could easily classify the running action from the other two actions due to the significant variation in the vibration of the legs. Fig.5 (b demonstrates the case where the system should recognize the walking action, which is similar to swinging arms to some extent. Due to the significant difference between walking and running, SMTRNN exclude the possibility of running quickly (before 7th frame). However, SMTRNN gets confused between walking and swinging arms. Therefore, the stimulus value corresponding to both the actions are high (close to 0.5) till the 19th frame after which the system can successfully classify the input signal.

Table 1 shows the recognition performance of the proposed model in recognizing single actions using our dataset. We define the highest output node as the real time classification output of the given input sample.

To test the robustness of the proposed SMTRNN model in classifying multiple actions we performed an experiment where the input signal was 3 times the length (270 frames) of a single action signal employed previously. In other words, the test signal included 3 different actions. It is to be noted that the SMTRNN was not trained with multiple actions signals. However, the system tried to use its knowledge based on single motion to classify the signals that contained multiple actions. The human subject changed his motion from first to second around 160 ~ 180thframes. Fig. 6 (a) shows a near-ideal output without any noise. We would like to emphasize that data used in Fig. 6 (a) was also used in Fig. 2 (d). The walking curve starts to decrease when human tester begins to change his motion from walking to swinging arms (in Fig. 6 (a)). There is a short delay (about 20 ~ 30 frames) until SMTRNN switches its output to the correct ones in all the 3 cases. Fig. 6 (b) and (c) indicates some trouble in classification between walking and swinging arms at first, but soon SMTRNN produced the correct output after about 20 frames. It's hard to define a boundary frame between the two actions, thus we cannot measure the recognition rate accurately. But we can still find that there is a clear cross point nearby 180th frame in Fig. 6 (a) ~ (c).

Table 1. Recognition result of 3 kinds of action

Human action	Recognition result %
Walking	96.4
Running	98.9
Swinging	97.6

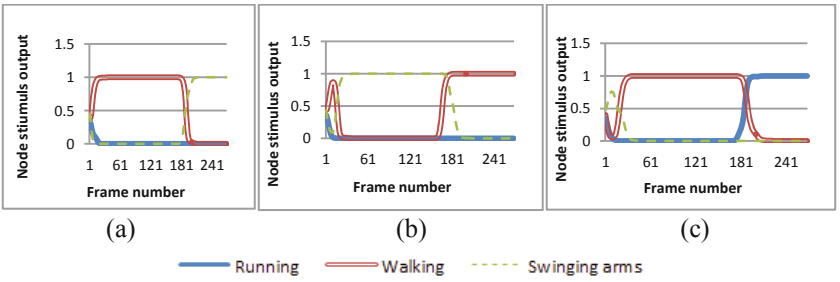


Fig. 6. Classification result of 3 continuous signals. (a)Walking +Swinging arms, (b) Swinging arms + Walking, (c) Walking +Running

5 Conclusion

In this paper, we proposed a dynamic classification model called SMTRNN for human action recognition and classification. The method was evaluated by experiments using KINECT. We investigated the possibility of slow context based classification. Although some certain slow nodes contain valuable information for classification, such information is weak and not stable for application. We conclude that our model can output a real time classification signal with high accuracy. It can also predict the

changes in human actions. Additionally, our model offers a possible way to perform a real-time human-computer interaction.

Acknowledgment. This work was supported by the Industrial Strategic Technology Development Program (10044009) funded by the Ministry of Knowledge Economy (MKE, Korea) (50%) and also supported by the Converging Research Center Program funded by the Ministry of Education, Science and Technology (2013K000333) (50%).

References

1. Arie, H., Arakaki, T., Sugano, S., Tani, J.: Imitating others by composition of primitive actions: A neuro-dynamic model. Robotics and Autonomous Systems 60(5), 729–774 (2012)
2. Cruz, L., Djalma, L., Luiz, V.: Kinect and RGBD Images: Challenges and Applications Graphics. In: 2012 25th SIBGRAPI Conference on Patterns and Images Tutorials (SIBGRAPI-T), August 22-25 (2012)
3. Doya, K., Yoshizawa, S.: Adaptive neural oscillator using continuous-time back-propagation learning. Neural Network 2, 375–386 (1989)
4. Funahashi, K., Nakamura, Y.: Approximation of dynamical systems by continuous timerecurrent neural networks. Neural Networks 6(6), 801–806 (1993)
5. Hinoshita, W., Arie, H., Tani, J., Okuno, H.G., Ogata, T.: Emergence of hierarchical structure mirroring linguistic composition in a recurrent neural network. Neural Networks 24(4), 311–320 (2011)
6. Husken, M., Stagge, P.: Recurrent Neural Networks for Time Series Classification. Neuro Computing 50(C), 223–235 (2003)
7. Jeong, S., Arie, H., Lee, M., Tani, J.: ·Neuro-robotics study on integrative learning of proactive visual attention and motor behaviors. Cogn. Neurodyn. 6, 43–59 (2011, 2012)
8. Joslin, C., El-Sawah, A., Chen, Q., Georganas, N.: Dynamic Gesture Recognition. In: IMTC 2005 – Instrumentation and Measurement Technology Conference, Ottawa, Canada, May 17-19 (2005)
9. Poppe, R.: A survey on vision-based human action recognition. Image and Vision Computing 28(6), 976–990 (2010)
10. Sakai, K., Kitaguchi, K., Hikosaka, O.: Chunking during human visuomotor sequence learning. Exp. Brain Res. 152, 229–242 (2003)
11. Tani, J., Nishimoto, R., Paine, R.: Achieving "organic compositionality" throughself-organization: reviews on brain-inspired robotics experiments. NeuralNetworks 21, 584–603 (2008)
12. Thomas, K., Andre, B., Michalis, F., KcC: HMM-based Human Motion Recognition with Optical Flow Data. In: 9th IEEE-RAS International Conference on Humanoid Robot, Paris, France, December 7-10 (2009)
13. Thoroughman, K.A., Shadmehr, R.: Learning of action through adaptive combination of motor primitives. Science 407, 742–747 (2000)
14. YamashitaY., T.J.: Emergence of functional hierarchy in a multiple timescaleneural network model: a humanoid robot experiment. PLoS ComputationalBiology 4(11) (2008)
15. Zhang, Y., Ogata, T., Takahashi, T., Okuno, H.G.: Dynamic Recognition of Environmental Sounds with Recurrent Neural Network. In: The 28th Annual Conference of the Robotics Society of Japan, Nagoya, Japan (2010)

Nonnegative Source Separation with Expansive Nonlinearity: Comparison with the Primary Visual Cortex

Hiroki Yokoyama

Graduate School of Information Systems,
The University of Electro-Communications,
1-5-1 Chofugaoka, Chofu, 182-8585 Tokyo, Japan
h-yokoyama@hi.is.uec.ac.jp

Abstract. We introduced the kernel trick to a linear generative model. In the present study, we trained a single layer model with nonnegativity constraint and expansive nonlinearity. After training, we found that the basis images acquired from natural scenes represented Gabor-like features. Moreover, the distributions of shape parameters of the basis images were similar to those found in V1. Other similar models, such as the sparse coding and the independent component analysis, fail to exhibit these properties.

Keywords: primary visual area (V1), simple cell, unsupervised learning, sparse coding, NMF, kernel trick.

1 Introduction

Many probabilistic models, such as the sparse coding[5], have provided insightful explanation of how cortical connectivity is organized to adapt the environment. Basic viewpoint of these models are that neural population in higher areas attempt to predict input signals from lower areas. Assuming the error between the prediction $\hat{\boldsymbol{x}}$ and the actual input $\boldsymbol{x}$ is normally distributed, the goal of this model is to minimize the sum of the squared error:

$$\begin{aligned} E &= \frac{1}{2}\sum_k ||\boldsymbol{x}_k - f(A\boldsymbol{y}_k)||^2 \\ &= \frac{1}{2}\mathrm{Tr}\left\{(X - f(AY))^T(X - f(AY))\right\} \end{aligned} \tag{1}$$

where T denotes the transpose of a matrix, $||\cdot||^2$ the squared norm in the Euclidean space, $Y = (\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots)$ represents the response of the neuronal population, $A = (\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots)$ the set of the receptive fields, $f(\cdot)$ a certain nonlinearity, and k is the index of the input image. The gradient of E gives the dynamics of the population response:

$$\Delta Y \propto -\frac{\partial E}{\partial Y} = A^T\left\{f'(AY) \otimes (f(AY) - X)\right\} \tag{2}$$

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 204–211, 2013.

where $\otimes$ denotes component-wise multiplication. In this model, the prediction $f(AY)$ is conveyed via the feedback connection A followed by the nonlinearity $f(\cdot)$, whereas the residual error is conveyed via feedback connection A^T.

On the other hand, many constructive models employ nonlinearities which conveys approximately the maximum values of the responses of the lower modules to the higher modules (e.g.[2]). This property seems to be important to learn the invariant response of complex cells. In this study, we use the kernel trick to introduce such nonlinearities to the modules of the model.

2 Model

We consider a function $\phi(X) = (\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \cdots)$ which maps vectors in the space of the visual input into a higher dimensional feature space. Measuring the distance between the prediction and the actual input in the feature space, we obtain the following optimization problem:

$$\min_{Y,A} E = \tfrac{1}{2}\mathrm{Tr}\left\{(\phi(X) - \phi(A)Y)^T(\phi(X) - \phi(A)Y)\right\} \tag{3}$$

$$s.t. \qquad Y \geq 0, A \geq 0, \forall i, ||\boldsymbol{a}_i||^2 = 1. \tag{4}$$

Then, instead of defining $\phi(\boldsymbol{x})$, we directly define the inner product of two vectors in the feature space:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{x})^T\phi(\boldsymbol{y}) \tag{5}$$

which is called kernel function.

If we set $k(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}^T\boldsymbol{y})$, we obtain the gradient of (3):

$$\Delta Y \propto -\frac{\partial E}{\partial Y} = f(A^TX) - f(A^TA)Y, \tag{6}$$

which shows that the input is conveyed via feedforward connection A^T followed by the static nonlinearity $f(s)$. To be physiologically plausible, we restrict $f(s)$ to be positive.

3 Numerical Experiment

To investigate the effects of the kernel functions, we trained the models with natural images.

The input images were ten $512 \times 512 - pixel$ gray scale photographs of natural scenes provided by Olshausen et al.[5]. These images were preprocessed by filtering with bandpass filter $R(f) = fe^{-(f/200)^4}$, which approximately corresponds to receptive fields in LGN. We extracted 9610 16×16-pixel image patches from the scenes and reshaped them into 256-dimensional column vectors. Here we define $X1$ as the matrix which consists of the column vectors. To simulate the responses of on- and off-channel receptive fields, we set $X_+ = (|X_1| + X_1)/2$

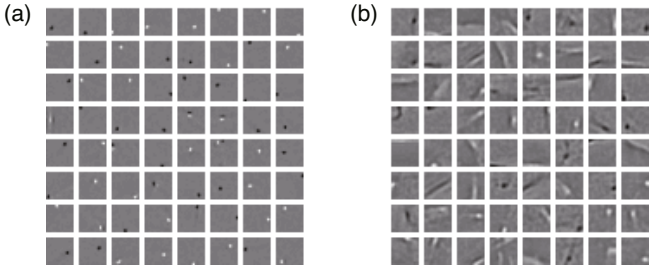


Fig. 1. The basis images acquired by training. (a) $\gamma = 1.1$. (b) $\gamma = 3.0$.

and $X_- = (|X_1| - X_1)/2$, $X = (X_+^T, X_-^T)^T$, and normalized its columns. The input X, described in section 2, was thus 512×9610 matrix with nonnegative elements.

We used multiplicative update rules[4] to update the model parameters. First, on the basis of the gradient of eq. (3), we obtained an additive update rule for Y:

$$Y \leftarrow Y + \eta \otimes (f(A^T X) - f(A^T A)Y). \tag{7}$$

Subsequently, we set $\eta = Y \oslash B$ and obtained

$$Y \leftarrow Y \otimes f(A^T X) \oslash f(A^T A)Y \tag{8}$$

where $\oslash$ denotes componentwise division. For A, we followed the same procedure to obtain the multiplicative update rule:

$$A \leftarrow A \otimes X(Y \otimes f'(A^T X))^T \oslash A(YY^T \otimes f'(A^T A)). \tag{9}$$

In above equations, the componentwise multiplication and division have lower precedence than matrix multiplication. We alternately updated Y and A using eqs. (8) and (9), respectively.

4 Result

4.1 Our Model

In this section, we discuss the basis images acquired by the training. Since the basis vectors, as well as the input vector, represent on- and off-channels separately, the nonnegative representation of image feature is not unique. For example, a certain pixel value $v > 0$ can be represented not only by on-channel v but also by combination of on-channel $v + w$ and off-channel w. For the acquired basis vectors, however, either of corresponding on- and off-channel was approximately zero. Therefore the basis images can be reconstructed by subtracting off-channel basis vector from on-channel.

Fig. 1 shows examples of the acquired basis images reconstructed as described above. (a) and (b) correspond to the cases where $\gamma = 1.1$ and $\gamma = 3.0$, respectively. When kernel function's exponent γ was small, the basis images represented

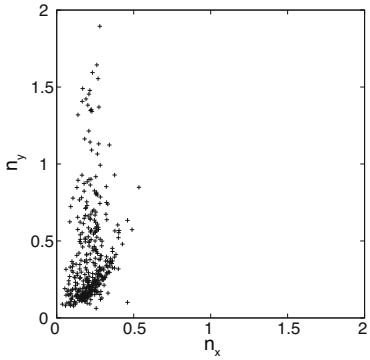


Fig. 2. Distribution of shapes of the basis images in the n_x–n_y plane ($\gamma = 2.5$)

single bright or dark spots. On the other hand, some of them became elongated and Gabor-like as γ increases. Most of them were Gabor-like for $\gamma \geq 2.5$. Therefore, when the nonlinearity is sufficiently expansive, our model can extract Gabor-like features.

4.2 Comparison with Related Models

Many other theories, such as the SC[5] and the independent component analysis (ICA)[1], also extract Gabor-like functions from natural scenes. However, there are subtle differences in detail between their result and actual receptive fields in V1. Ringach[6] reported that both of these theories predict receptive fields with a larger number of subfields than those in the experimental data. In addition, they do not generate receptive fields that are broadly tuned in orientation, which are commonly seen in the monkey V1. Here we compare our model's basis images with these theories and the experimental data.

To analyze the shape of basis images, a two-dimensional Gabor function

$$g(x', y') = A \exp\left(-\left(\frac{x'^2}{2\sigma_{x'}^2} + \frac{y'^2}{2\sigma_{y'}^2}\right)\right) \cos(2\pi f x' + \phi) + B \tag{10}$$

$$x' = (x - x_0)\cos\theta + (y - y_0)\sin\theta \tag{11}$$

$$y' = -(y - y_0)\sin\theta + (y - y_0)\cos\theta \tag{12}$$

was fit to our result. The coordinate system (x', y') is obtained by translating the original coordinate system (x, y) by (x_0, y_0) and rotating it by θ. In this coordinate system, the x' axis represents the direction the cosine function varies. The parameters $\sigma_{x'}$ and $\sigma_{y'}$ represent the width of the Gaussian envelope along the corresponding axes, f and ϕ the spatial frequency and the spatial phase respectively. The parameters A and B are the amplitude and the constant, respectively, which are not important for the shape properties but are required for the fitting.

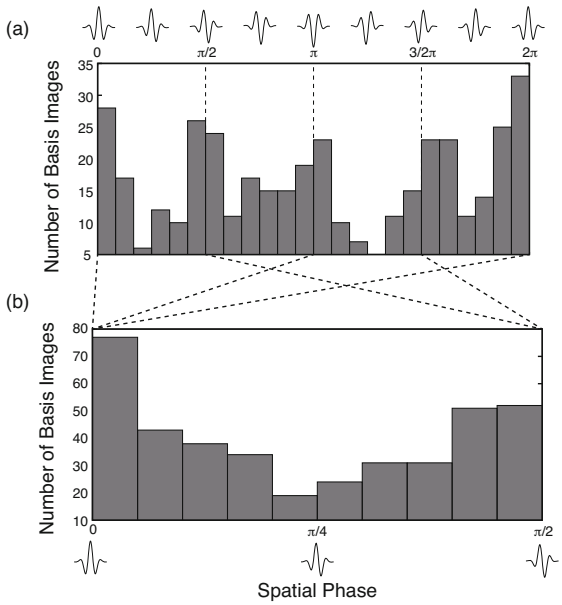


Fig. 3. Histograms of the spatial phase parameters. (a) The original spatial phase ϕ. (b) The symmetry parameter $\hat{\phi}$ obtained by eq. (13).

Ringach[6] compared SC and ICA with the experimental data by the numbers of subfields in the receptive fields. This property is obtained as $n_x = \sigma_{x'} f$ and $n_y = \sigma_{y'} f$, which can be considered as the numbers of sinusoidal cycles fit into a segment of length $\sigma_{x'}$ and $\sigma_{y'}$, respectively. Receptive fields are circular when $n_x = n_y$, have many subfields when n_x is large, and are blob-like when (n_x, n_y) is located near the origin. His study showed that the distribution of (n_x, n_y) lay approximately on a one-dimensional curve, which passed near the origin. The basis images obtained by SC and ICA, on the other hand, tend to have more subfields.

The relationship between n_x and n_y for our model is shown in Fig. 2. This shows that the population includes both blob-like and elongated Gabor-like shapes. In addition, it shows also that a part (at the lower right) of the population appeared to form a curve similar to that found in the experimental data.

The spatial phase properties of receptive fields are also discussed in [6]. This study reported that the receptive fields clustered into even and odd symmetry classes. Interestingly, there was a tendency for receptive fields that are "well tuned" in orientation and spatial frequency (with (n_x, n_y) located away from the origin) to be odd symmetric and, conversely, for those that are "broadly tuned" (with (n_x, n_y) located near the origin) to be even symmetric. To analyze the symmetry of basis images for a comparison, we employed the symmetry parameter defined in [6]:

$$\hat{\phi} = \arg(|\cos\phi| + i|\sin\phi|), \tag{13}$$

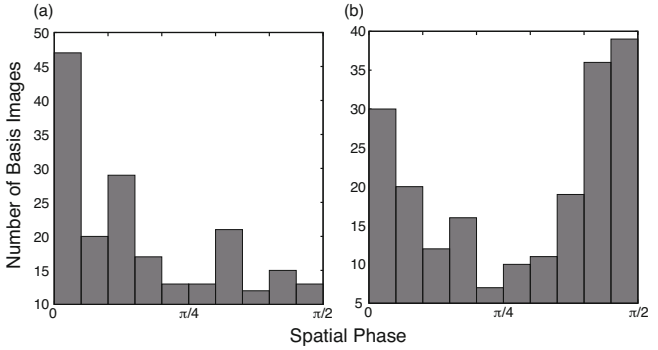


Fig. 4. Histograms of the symmetry parameters for (a) broadly tuned and (b) well tuned basis images

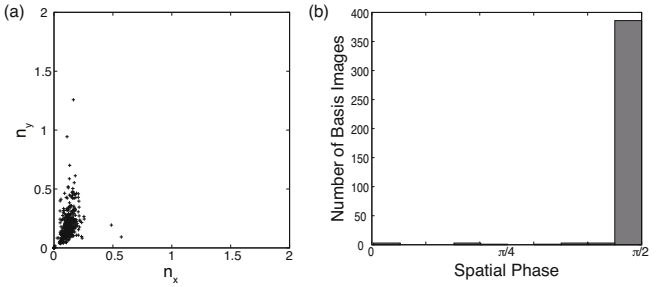


Fig. 5. The shape properties of nonnegative matrix factorization with sparseness constraints[3]. (a) A scatter plot in the n_x–n_y plane, corresponding to Fig. 2. (b) A histogram of symmetry parameter $\hat{\phi}$, corresponding to Fig. 3(b).

where i denotes imaginary unit. In this notation, the Gabor function is even symmetric when $\hat{\phi} = 0$ and odd symmetric when $\hat{\phi} = \pi/2$.

Fig. 3(a) is a histogram of the spatial phases of the basis images. It shows that the distribution of the spatial phases is not uniform, and many of the basis images are either even- or odd- symmetric, which can be seen more clearly by obtaining $\hat{\phi}$, which is shown in Fig. 3(b).

Fig. 4(a) and (b) are histograms of the symmetry parameters for broadly tuned and well tuned basis images, respectively. Fig. 4(a) shows that broadly tuned basis images tend to be even symmetric, and Fig. 4(b), compared with Fig. 3(b), indicates a tendency for well tuned ones to be odd symmetric.

In addition to SC and ICA, the nonnegative matrix factorization (NMF) also extracts Gabor-like features when sparseness constraint is added (Hoyer, 2004[3]). To compare the shape properties of Hoyer's and our models, we trained Hoyer's model with the same set of input images as that used above.

Fig. 5(a) shows the shape distribution of the basis images in the n_x–n_y plane. It can be seen from this distribution that this model generates not only well tuned but also broadly tuned filters, as well as our model. On the other hand,

this model does not exhibit the spatial phase property. Fig. 5(b) shows that almost all the basis images are odd symmetric ($\hat{\phi} \approx \pi/2$).

5 Conclusion

The sparse coding and other generative models have provided explanations of how the visual system works (e.g., the role of bidirectional connections and how the Gabor-like receptive fields are acquired). However, the response properties derived from minimization of linear prediction error does not always agree with the static nonlinearity of cortical neurons. In the present study, we introduced nonlinearity to the generative model by representing input and basis images as elements of a feature space. In addition, we constrained the model parameters to be nonnegative for the sake of physiological plausibility.

After training with natural images, our model exhibited properties similar to V1 neurons, e.g. sparse response, Gabor-like receptive fields. Furthermore, the distributions of shape parameters of acquired basis images were more similar to those found in V1 than other models. The shape properties should be an important part of difference between our model and other models. The comparison with other models suggests that the nonnegative source separation (both Hoyer's and our models) can extract both blob-like and Gabor-like features from natural scenes, unlike SC or ICA. However, the nonnegativity is not sufficient for the spatial phase property. Our model, with expansive nonlinearity, can extract both even and odd symmetric features, whereas Hoyer's can extract only odd symmetric features. This suggests that both nonnegativity and nonlinearity are necessary to exhibit spatial properties similar to receptive fields in V1.

The question then arises: what is the statistical role of the nonlinearity? Although it requires further study to answer this question, we can consider a probabilistic model including the nonlinearity as follows. The optimization function can be regarded as the negative logarithm of the posterior of A and Y, namely,

$$p(A, Y|X) \propto \exp\left(-\mathrm{Tr}\left(f(X^{\mathrm{T}}X) - 2f(A^{\mathrm{T}}X)^{\mathrm{T}}Y + f(A^{\mathrm{T}}A)YY^{\mathrm{T}}\right)\right). \quad (14)$$

In this interpretation, the solution to the optimization corresponds to maximum *a posteriori* (MAP) estimation of A and Y. Assuming A is constant, eq. (14) means that the posterior distribution of Y is truncated normal distribution (the truncation is due to nonnegativity of Y), whose parameters depend nonlinearly on X. The posterior distribution of A can be considered as well, although its shape is more complicated because of the nonlinearity. On the other hand, the model of input images $p(X|A, Y)$ which is consistent with the posterior $p(A, Y|X)$ is not Gaussian, because of the nonlinearity. Therefore, although our model exhibit similar properties to SC and other linear models, these are essentially different as probabilistic models.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Number 24008269.

References

1. van Hateren, J.H., Ruderman, D.L.: Independent component analysis of natural image sequences yields spatiotemporal filters similar to simple cells in primary visual cortex. Proceeding of the Royal Society B: Biological Sciences 265(1412), 2315–2320 (1998)
2. Heeger, D.J.: Normalization of cell responses in cat striate cortex. Visual Neuroscience 9(2), 181–197 (1992)
3. Hoyer, P.O.: Non-negative matrix factorization with sparseness constraints. Journal of Machine Learning Research 5, 1457–1469 (2004)
4. Lee, D.D., Seung, H.S.: Algorithms for nonnegative matrix factorization. In: Advances in Neural Information Processing, ch. 13. MIT Press (2001)
5. Olshausen, B.A., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381(6583), 607–609 (1996)
6. Ringach, D.L.: Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. Journal of Neurophysiology 88(1), 455–463 (2002)

Deep Relational Machines

Huma Lodhi

School of Engineering, University of Warwick, Coventry, CV4 7AL, UK
huma.m.lodhi@gmail.com

Abstract. Deep learning methods that comprise a new class of learning algorithms give state-of-the-art performance. We propose a novel methodology to learn deep architectures and refer to it as a deep relational machine (DRM). A DRM learns the first layer of representation by inducing first order Horn clauses and the successive layers are generated by utilizing restricted Boltzmann machines. It is characterised by its ability to capture structural and relational information contained in data. To evaluate our approach, we apply it to challenging problems including protein fold recognition and detection of toxic and mutagenic compounds. The experimental results demonstrate that our technique substantially outperforms all other approaches in the study.

Keywords: Deep Relational Learning, Restricted Boltzmann Machines, Inductive Logic Programming, Multi-class classification.

1 Introduction

Deep learning (DL) is an emerging state-of-that-art class of algorithms, and whose well-known examples are deep belief networks (DBNs) [1] and stacked auto encoders (SAEs) [2]. In DL, a machine learns many layers of representation to capture structure and variations in data, and this process generally improves its generalisation performance.

The standard DL techniques operate on input features (derived from raw input data) that are in binary or numeric form. The data generated in many real-world problems, and especially in biological and chemical domains, is naturally relational. The conversion of data into a form that is amenable to such techniques may loose important information during this process. This establishes a need to propose a novel DL methodology that can handle arbitrary forms of data. First order logic (FOL) provides a useful means to represent data and is well known for its expressive powers. McCarthy who has pioneered the application of logic to construct intelligent machines has observed "If one wants a machine to be able to discover an abstraction, it seems most likely that the machine must be able to represent this abstraction in some relatively simple way", and FOL has been viewed as a means to increasing generality, and expressing knowledge and reasoning [3]. In FOL we can efficiently express both complex information that is needed for solving complex tasks like chess problems and commonsense

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 212–219, 2013.

knowledge that is required to do a simple task, hence in this paper we take a logicist approach for learning deep relational architectures, namely, deep relational machines (DRMs).

A DRM learns a deep architecture by stacking a set of induced logical rules and any number of generative models. It learns the first layer of representation by using an Inductive Logic Programming (ILP) system. A set of hypothesised rules are used as an input for the second layer, where a restricted Boltzmann machine (RBM) is used for learning the second and successive layers. Once a suitable representation is obtained, classification or regression problems are solved by using support vector machines (SVMs) in conjunctions with the features learned at the highest layer. In DRM the use of first order logic provides the flexibility of using arbitrary forms of structured and non-structured data, and it exploits semantic and syntactic relational structures and variations to train a machine with low error probability.

ILP has been used to construct features and this process is termed as propositionalisation [4]. Recently, features (rules) generated by ILP algorithms have been used to design logic based kernel methods [5,6]. The design of a DRM and its objectives are fundamentally different from these apparently related approaches. It is based on the belief that the representation learned by an ILP system does not capture all the structure and variations in data, and much needed refinement is achieved by performing deep learning.

We evaluate the proposed methodology by applying it to biological and chemical domains, namely protein fold recognition and identification of toxic and mutagenic compounds. The experimental results show that our approach provides accurate solutions to complex problems and generally outperforms related approaches.

2 Generative Models and Inductive Logic Programming

Generative Models: RBM [7] is a well known example of generative models. It displays a bipartite structure and consists of a single hidden layer where there are connections between the units of visible and hidden layers but there are no connections between the units of any individual layer. The connections between the units are weighted and there is also a bias associated with each unit. In its standard form, the hidden and visible units of an RBM are of the form $(0, 1)$. The probability over the joint configuration of hidden and visible units is given by $p(\mathbf{d}, \mathbf{h}) = \frac{exp\Big(-\mathcal{E}\big(\mathbf{d},\mathbf{h}\big)\Big)}{Z}$, where $\mathbf{d}$ and $\mathbf{h}$ are input and hidden vectors. Z is a normalization constant described by the expression: $Z = \sum_{\mathbf{d},\mathbf{h}} exp\Big(- \mathcal{E}\big(\mathbf{d}, \mathbf{h}\big)\Big)$. $\mathcal{E}$ is an energy function and is defined by $\mathcal{E}(\mathbf{d}, \mathbf{h}) = -\mathbf{c}^\intercal\mathbf{h} - \mathbf{b}^\intercal\mathbf{d} - \mathbf{h}^\intercal\mathbf{W}\mathbf{d}$. In this expression the vector $\mathbf{h}$ gives the hidden layer activations and the activations of visible units (input data) is supplied by $\mathbf{d}$. The entries of vector $\mathbf{b}$ are indexed by the biases of visible units and $\mathbf{c}$ is a bias vector for hidden layer. $\mathbf{W}$ is $N * M$ matrix where N represent numbers of visible units

and the number of hidden units is denoted by M. The training of an RBM can be performed by maximizing the log-likelihood using a gradient ascent scheme. As the computations can become intractable, an approximation procedure is used. In this method, data is sampled back and forth and error is minimized.

Inductive Logic Programming. The class of learning algorithms that form ILP are well known for representing relations and structures in data. In ILP first order logic is used to express data, background knowledge (prior knowledge) and the induced hypothesis. The standard learning framework, within which most ILP algorithms are designed, is learning from entailment. In this setting, we consider a training set $D = \{(\mathbf{d}_1, t_1), (\mathbf{d}_2, t_2), \ldots, (\mathbf{d}_n, t_n)\}$ of input-output pair examples and background knowledge B. Each $\mathbf{d}_i$ belongs to a domain $\mathcal{D}$ and each t_i belongs to the set T, where $T = \{0, 1\}$. All the examples having $t_i = 1$ form the positive training set D^+, and the negative training set D^- comprises all the examples having $t_i = 0$. An ILP algorithm aims to induce a hypothesis (set of Horn clauses, rules) $\mathfrak{F}$ from background knowledge, B, and data D with the aim that it does not imply or more specifically entail any negative example, and all the positive examples satisfy the conditions of the hypothesis. Formally, $B \cup \mathfrak{F} \models D^+$ and $B \cup \mathfrak{F} \not\models D^-$.

In this paper we use the symbol $\mathfrak{F}$ for a hypothesis (typically a set of a few rules) returned by a standard ILP algorithm, the symbol $\mathbb{F}$ for a hypothesis that is comprised of all the R rules generated during the search of the hypothesis space and symbol F for a hypothesis that is given by a set of N rules that describe the first layer of a DRM.

3 Deep Relational Machines

The Logical Layer. We build the first layer by adopting the learning from entailment setting and term it the logical layer. The data D comprising positive and negative examples are encoded as ground facts. The background knowledge B is given by a set of Horn clauses where the set comprises ground facts or non-ground clauses. A hypothesis F, in the form of Horn Clauses (rules), is induced. The data is transformed for the next layer by determining whether an example $\mathbf{d}_i$ is implied by the hypothesis F conjoined with the background knowledge. A rule $f_j \in F$ conjoined with background knowledge B is viewed as an encoding function $E(\mathbf{d}, B, F)$ that maps data into the Boolean representation as described below:

$$E(\mathbf{d}_i, B, f_j) = \begin{cases} 1 & \text{if } B \cup f_j \models \mathbf{d}_i \\ 0 & \text{if } B \cup f_j \not\models \mathbf{d}_i \end{cases}$$

In contrast to DBN and SAE, a DRM performs supervised learning in the logical layer. The methodology is presented as Algorithm 2. The algorithm takes as input data that is encoded in FOL form. It utilizes an ILP system to generate a set of first order logical rules and considers all the clauses generated during the search of the hypothesis space as candidate features. The set of induced features can be very very large but finite. In this paper we introduce a well known measure, namely Gain Ratio (GR) to score the rules. The selection of the rules

Algorithm 1. Quantification of the goodness of rules and selection of a subset

Input: A set of input-output pair examples D where $\mathbf{d}_i \in \mathcal{D}$ and $t_i \in \{0, 1\}$. The domain background knowledge B, a set $\mathbb{F}$ of R induced rules, and N
Output: A set of selected rules F
for $j = 1$ to R **do**
Compute gain ratio value for each rule f_j that quantifies its goodness
end for
Sort the computed values and select the first N rules with highest GR values.
Return: $F = \{f_1, f_2, \ldots, f_N\}$

Algorithm 2. The logical layer for DRM for binary classification

Input: A set of input-output pair training examples $D = \{(\mathbf{d}_1, t_1), (\mathbf{d}_2, t_2) \ldots, (\mathbf{d}_n, t_n)\}$ where $\mathbf{d}_i \in \mathcal{D}$ and $t_i \in \{0, 1\}$
Output: An encoding function E that maps data into the Boolean representation $E : \mathbf{d}_i \rightarrow (\alpha_j)_{j=1}^{j=N}$ where $\alpha_j = 0$ or $\alpha_j = 1$
Induce a set of rules $\mathbb{F}$ by using an ILP algorithm.
Consider all the R rules generated during the search of the hypothesis space.
Select the most informative subset $F = \{f_1, f_2, \ldots, f_N\}$ by using Algorithm 1.
Compute coverage by using entailment: $E(\mathbf{d}_i, B, F) = (\alpha_1, \ldots, \alpha_N)$, where
if $f_j \cup B \models \mathbf{d}_i$ **then**
$\alpha_j = 1$
else if $f_j \cup B \not\models \mathbf{d}_i$ **then**
$\alpha_j = 0$
end if
Return $E(\mathbf{d}, B, F)$

is an important process as it reduces complexity and improves generalization performance of a DRM. To derive an expression for GR we first define entropy and Information Gain (IG). The entropy of a logically encoded (training) data D is the expected value of information required to assign a category to an example: $entropy(D) = H(D) = -\sum_{1 \leq \iota \leq m} \frac{|D_\iota|}{|D|} \log_2 \left(\frac{|D_\iota|}{|D|} \right)$. $|D_\iota|$ is the number of examples that are grouped into category ι. Information gain measures expected reduction in entropy. The IG value between a rule $f_j$and a category ι is computed by considering the division of data D into subsets according to the varying values of the rule, and it is given as: $IG(f_j) = H(D) - \sum_{i \in values(f_j)} \frac{|D_i|}{|D|} H(D_i)$. We obtain a mathematical expression for GR by normalising IG. The normalization value (Z) is given by: $Z(f_j) = -\sum_{i \in values(f_j)} \frac{|D_i|}{|D|} \log_2 \left(\frac{|D_i|}{|D|} \right)$. Now we can define the gain ratio of the rule as follows:$GR(f_j) = \frac{IG(f_j)}{Z(f_j)}$. The method computes GR values for all rules, and identifies the most relevant and informative features by using the procedure described as Algorithm 1. As the higher values of GR tells the goodness of fit for a rule, the N rules with the highest GR are selected.

Algorithm 3. The logical layer for DRM for multi-class classification

Input: A set of input-output pair training examples $D = \{(\mathbf{d}_1, t_1), (\mathbf{d}_2, t_2) \ldots, (\mathbf{d}_n, t_n)$ where $\mathbf{d}_i \in \mathcal{D}$ and $t_i \in \{1, 2, \ldots, m\}$. The background domain knowledge B
Output: An encoding function E that maps data into the Boolean representation $E : \mathbf{d}_i \rightarrow (\alpha_j)_{j=1}^{j=N}$ where $\alpha_j = 0$ or $\alpha_j = 1$
$F = \{\}$
for $\iota = 1$ to m **do**
Select a class from m classes .
Formulate the binary class problem by assigning label '1' to examples of class p and '0' to examples of remaining classes.
Generate a set of rules $\mathbb{F}_\iota$ by inducing an ILP algorithm.
Consider all the R_ι rules generated during the search of the hypothesis space.
Select the most informative and relevant subset F_ι by invoking Algorithm 1.
Add F_ι to F.
end for
The final F is given by: $F = \{F_1, F_2, \ldots, F_m\}$.
Assess whether examples are implied by rules $f_j \in F$: $E(\mathbf{d}_i, B, f_j) = (\alpha_1, \ldots, \alpha_N)$.
Return: $E(\mathbf{d}, B, F)$

The logical layer of a DRM is learned by using the procedure described as Algorithm 2 for examples belonging to two classes. In scenarios where examples belong to $m > 2$ categories, we propose a strategy given as Algorithm 3 to construct the logical layer. The goal is to extract relevant features that describe the data belonging to diverse classes. The multi-class problem is divided into m binary tasks and a positive training set is defined by selecting a class ι where the examples belonging to the remaining classes form the negative training set. A hypothesis that consists of N rules is induced. A subset F_ι is selected by applying Algorithm 1 to a validation or training set. The process is repeated for all the classes, and hence comprises m iterations. The obtained m subsets are merged into a rule set F. If there are any redundant rules they are removed from the set F. An input data vector for the next layer is generated by assessing whether the example satisfy the conditions of the rule set F or not.

The Generative Layers. As the features, learned in the first layer, may not model all the structure, relations and variations in the data, a DRM learns more layers of representation by using RBMs. In these layers an RBM is trained in an unsupervised manner: the weights and the biases are measured, activations of input and hidden unit are computed. The activations of the hidden units of the previous layer become the input data for the current RBM. This process is repeated l times. Predictive problems are solved by using the extracted features.

4 Experiments and Analysis

In this section we describe experiments and results. We empirically tested the proposed method on datasets comprising protein domains, mutagenic and toxic

Table 1. 10-fold cross-validated accuracy ± standard deviation for mutagenesis

kFOIL	nFOIL	PROGOL	SVILP	DRM(1)	DRM(4)
81.3 ± 11.0	75.4 ±12.3	78.7 ± 10.8	87.2 ± 07.5	88.82 ± 7.63	**89.90 ± 6.33**

Table 2. Five-fold cross validated accuracy ± standard deviation for DSSTOX dataset for MC_ILP, DL_SVILP and DRM (DRM(1) & DRM(4))

MC_ILP	DL_SVILP	DRM(1)	DRM(4)
57.1 ± 2.4	65.2 ± 02.3	63.57 ± 2.29	**66.06 ± 2.25**

compounds. The experiments were performed by a combination of in house scripts and publicly available software systems. A deep relational architecture was learned that comprised of four layers. The first layer was given by hypothesised rules, and the remaining layers of representation were learned by an RBM. The activation function for these layers was a logistic sigmoid. The features generated at the first and the fourth layer were used in conjunction with an SVM to perform classification tasks. A DRM with one layer and four layers is denoted by DRM(1) and DRM(4) respectively. We compared the performance of DRM with SVILP, PROGOL, multi-class ILP (MC_ILP) [8], and decision list based SVILP (DL_SVILP) [9]. We used accuracy [10] as the evaluation measure. For multi-class classification problems we calculated accuracy for each class and for overall (OA) of the dataset.

Datasets: We conducted experiments on three benchmark datasets. The mutagenesis dataset [11] comprises 188 compounds. Of the 188 molecules, 125 are positive examples and 63 are negative examples. The EPA Fathead Minnow Acute Toxicity database (DSSTox) contains highly diverse organic compounds [12]. In the database there are 442 compounds with unique mode of action that is experimentally determined. The compounds are placed into 8 categories, and the class distribution is skewed in this dataset. Protein dataset [13] has been extracted from the Structural Classification of Proteins (SCOP) database. It contains protein domains belonging to forty five folds (classes). These folds are categorized into four main structural classes, namely α, β, α/β and $\alpha + \beta$. The number of test protein domains are 405. The distribution of protein folds in the dataset is skewed where number of domains in a fold ranges from four to thirty two.

In the case of mutagenesis dataset molecules were represented by using the key information that was given in the form of atom and bond. The compounds in DSSTox dataset were represented by atom bond descriptions, functional groups and rings. The background knowledge described in [13] was used to represent protein domains.

Results: The results presented in Table 1 demonstrated efficacy of DRM for mutagenesis dataset. DRM substantially improved performance of PROGOL and

Table 3. Average accuracy ± standard deviation for protein fold dataset for MC_ILP, DL_SVILP and DRM (DRM(1) & DRM(4))

Fold	MC_ILP	DL_SVILP	DRM(1)	DRM(4)
α (8 classes)	57.78 ± 5.21	62.22 ± 5.11	67.02 ± 4.85	**74.47 ± 4.50**
β (14 classes)	33.64 ± 4.57	45.79 ± 4.82	42.31± 4.84	44.23 ± 4.87
α/β (14 classes)	56.45 ± 4.45	62.90 ± 4.33	62.60 ± 4.36	**65.04 ± 4.30**
$\alpha + \beta$ 9 classes)	66.67 ± 5.41	72.62 ± 5.27	77.01 ± 5.19	**88.51 ± 5.11**
All (45 classes)	52.84 ± 2.48	60.25 ± 2.43	61.52 ± 2.41	**66.92 ± 2.33**

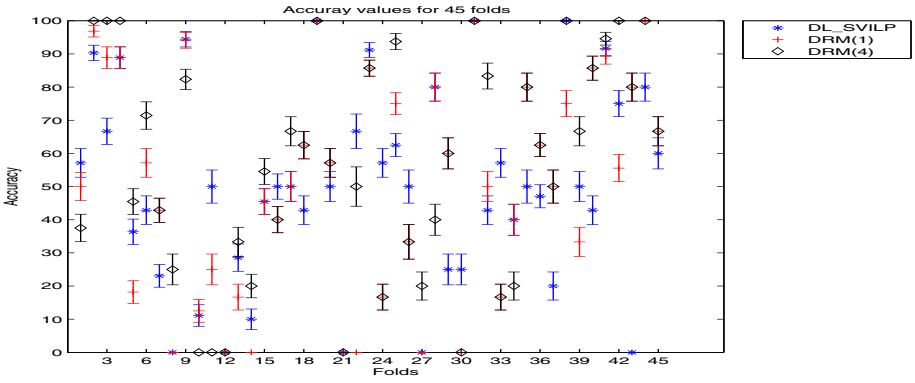


Fig. 1. Accuracy values for DL_SVILP and DRM (DRM(1) & DRM(4)) for 45 protein folds

SVILP. We also contrasted DRM with kFOIL and nFOIL by considering the results reported in [6]. The accuracy values confirmed its low error probability as compared to the related shallow methods. Five-fold cross-validated accuracy values were presented in Table 2 for the DSSTox dataset. We compared performance of proposed technique with MC_ILP and DL_SVILP. The results showed usefulness of deep learning in detecting toxic compounds. In Table 3 and Figure 1, the results of experiments on protein fold dataset were shown. From the results it was evident that DRM substantially and significantly improved upon MC_ILP and DL_SVILP. DRM improved upon DL_SVILP for 23 folds. It was worth noting that the over all accuracy of DRM was substantially and significantly better than DL_SVILP for α, α/β, and $\alpha + \beta$ structural classes. These experiments showed that DRM generally outperformed the related approaches in the study.

5 Conclusion

In this paper we have presented a novel methodology to learn a deep relational architecture by integrating hypothesised logical rules and generative models. The performance of the proposed technique is empirically tested by applying

it to complex tasks of protein fold recognition and classification of toxic and mutagenic compounds. Experimental comparisons of the performance of deep relational machine with related approaches show that the proposed approach generally achieves an accuracy that is substantially higher than all the other methods considered in the study. We aim to further develop DRM by extending the integration stage, exploring non-probabilistic methods for the second and the successive layers and applying it to other challenging real-world problems.

Acknowledgements. The author would like to thank David C Hogg and Tony Cohn for useful comments on draft of the paper. The author would also like to acknowledge the support of the FP7 Collaborative Project COGNITO.

References

1. Hinton, G.E., Salkhutdinov, R.: Reducing the dimensionality of data with Neural Networks. Science 33, 504–507 (2006)
2. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: Adv. in Neu. Infor. Processing Systems 19, pp. 153–160. MIT Press (2007)
3. McCarthy, J.: Programs with common sense. In: Proc. of Teddington Conf. on the Mechanization of Thought Processes, pp. 75–91. Her Majesty's Stationery Office, London (1959)
4. Kramer, S., Lavrac, N., Flach, P.: Propositionalisation approaches to Relational Data Mining. In: Dzeroski, S., Larac, N. (eds.) Relational Data Mining, pp. 262–291. Springer, Berlin (2001)
5. Muggleton, S., Lodhi, H., Amini, A., Sternberg, M.J.E.: Support Vector Inductive Logic Programming. In: Hoffmann, A., Motoda, H., Scheffer, T. (eds.) DS 2005. LNCS (LNAI), vol. 3735, pp. 163–175. Springer, Heidelberg (2005)
6. Landwehr, N., Passerini, A., Raedt, L., Frasconi, P.: kFOIL: Learning simple relational kernels. In: Proc. of the Nat. Conf. on Artificial Intelligence (AAAI), vol. 21, pp. 389–394 (2006)
7. Ackley, D.H., Hinton, G.E., Sejnowski, T.J.: A learning algorithm for Boltzmann machines. Cognitive. Sci. (9), 147–169 (1985)
8. Laer, W.V., de Raedt, L., Dzeroski, S.: On multi-class problems and discretization in Iductive Logic Programming. In: Proc. of the 10th Int. Symposium on Foundations of Intelligent Systems, pp. 277–286 (1997)
9. Lodhi, H., Muggleton, S., Sternberg, M.J.E.: Learning large margin first order decision lists for multi-class classification. In: Gama, J., Costa, V.S., Jorge, A.M., Brazdil, P.B. (eds.) DS 2009. LNCS, vol. 5808, pp. 168–183. Springer, Heidelberg (2009)
10. Ding, C.H., Dubchak, I.: Multi-class protein fold recognition using support vector machines and neural networks. Bioinformatics 17, 349–358 (2001)
11. Debnath, A.K., de Compadre, R.L.L., Debnath, G., Schusterman, A.J., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatics nito compounds. correlation with molecular orbital energies and hydrophobicity. J. Med. Chem. 34(2), 786–797 (1991)
12. Richard, A., Williams, C.: Distributed structure-searchable toxicity (DSSTox) public database network: A proposal. Mutat. Res. 499(1), 27–52 (2002)
13. Cootes, A.P., Muggleton, S., Sternberg, M.J.: The automatic discovery of structural principles describing protein fold space. J. Mol. Biol. 330(4), 839–850 (2003)

Online Incremental Structure Learning of Sum–Product Networks

Sang-Woo Lee, Min-Oh Heo, and Byoung-Tak Zhang

School of Computer Science and Engineering,
Seoul National University,
1 Gwanak-ro, Gwanak-gu, Seoul 151-744, Korea
{slee,moheo,btzhang}@bi.snu.ac.kr

Abstract. Sum–product networks (SPNs) are deep architectures that can learn and infer at low computational costs. The structure of SPNs is especially important for their performance; however, structure learning for SPNs has until now been introduced only for batch-type dataset. In this study, we propose a new online incremental structure learning method for SPNs. We note that SPNs can be represented by mixtures of basis distributions. Online learning of SPNs can be formulated as an online clustering problem, in which a local assigning instance corresponds to modifying the tree-structure of the SPN incrementally. In the method, the number of hidden units and even layers are evolved dynamically on incoming data. The experimental results show that the proposed method outperforms the online version of the previous method. In addition, it achieves the performance of batch structure learning.

Keywords: sum–product networks, structure learning, online learning, incremental learning, deep architecture, probabilistic graphical model.

1 Introduction

As learning the structure of graphical models is one of the most important issues in machine learning fields, many researchers have contributed to its study. Noteworthy examples of these research studies are Bayesian networks [1], Markov networks [2], deep networks [3], and sum–product networks (SPNs) [4].

Studies on online learning, however, are limited because it is not easy to change the form of probability tables without information about forgotten training data. Nevertheless, online learning is an essential problem in machine learning, and there are some learning environments in which it should be applied, such as large-scale data learning or lifelong learning. In a successful study, a single-layer denoising autoencoder was learned using online incremental structure learning [5]. It had been verified that this model learns the changing probability distribution of data. They also argued the possibility of extension to multi-layer models.

In order to solve the online learning problem using enough representation power, however, we focus on the SPN, a hierarchical architecture that includes sum nodes, product nodes, and univariate nodes [6]. In recent research on SPNs [4], Gens and Domingos used hierarchical biclustering to learn the structure of SPNs well.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 220–227, 2013.

The framework that they used is, however, not suitable for online learning because entire data should be used in the structure learning steps. Toward making an incremental model, we first note that structure learning of SPNs is highly dependent on clustering instances. Using this perspective, we convert the online structure learning problem to an online clustering problem. We propose a simple mini-batch clustering algorithm which can modify the number of clusters dynamically on incoming data, and apply it to incremental structure learning. The experiments show that it outperforms the online version of the previous method [4], and achieves the performance of batch structure learning.

The remainder of the paper is organized as follows: In section 2, we introduce a brief definition of SPNs, and a previous study on structure learning. In section 3, we suggest online incremental structure learning methods. In section 4, we show the experimental results of applying online incremental learning methods, and conclude this paper in section 5.

2 Sum–Product Networks

2.1 Representation of SPNs

SPNs are one of probabilistic graphical models (PGMs) which have specialized structure for fast inference. They are constrained to have tree structures (rooted directed acyclic graphs), and their leaves represent univariate distribution such as multinomial distribution for discrete variables, Gaussian, Poisson and other continuous distributions. Internal nodes in the tree represent products or sums of their children with the corresponding weights as the Figure 1. Recursive definition of SPNs introduced in [4] is as follows.

Definition 1. An SPN is defined as follows:

1. A tractable univariate distribution is an SPN.
2. A product of SPNs with disjoint scopes is an SPN.
3. A weighted sum of SPNs with the same scope is an SPN, provided all weights are positive.
4. Nothing else is an SPN.

The scope of an SPN is defined as the set of variables that appear in it. The sub-SPN S_i is a sub-tree at a node i as a root and corresponds to a probability distribution over its scope.

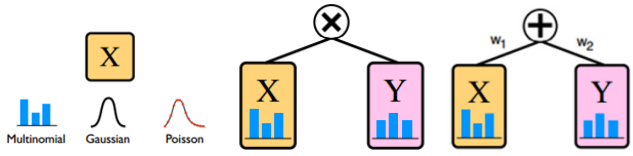


Fig. 1. Types of nodes in SPN: Univariate node (left); Product node (middle); Sum node (right)

Let $x_1, \ldots, x_n$ are variables in an SPN S and $x = (x_1, \ldots, x_n)$ be a state in their possible world. $S_i[x]$ be the probability of the i-th node of the state x, and $S_0[x]$ be the probability represented by the root node (the *0*-th node). Considering the weights, the recursive relationship among $S_i[x]$ of arbitrary nodes is as follows:

$$S_i[x^{(i)}] = \begin{cases} \sum_{j \in ch(i)} w_{ij} S_j[x^{(j)}] & , \quad if\ S_i\ is\ sum\ node \\ \prod_{j \in ch(i)} S_j[x^{(j)}] & , \quad if\ S_i\ is\ product\ node \\ c_i P_i(x^{(i)}) & , if\ S_i\ is\ univariate\ node \end{cases} \tag{1}$$

where $x^{(i)}$ represents the input variables that appear in a sub-SPN S_i, $ch(i)$ is the index set of child nodes of node i, and P_i is a generic univariate distribution of the variable, which $x^{(i)}$ contains only one variable. The joint probability distribution of an SPN S is $P(x) = S_0[x]$ if the weights at each sum node sum to one.

Many previous works related to SPNs, including sigma-pi neural networks [7], arithmetic circuits [8], and other compact representations exist. However, SPNs are a more general probabilistic model that enjoys enough representation power. SPNs show a remarkable performance in image classification tasks [9] and video learning [10].

2.2 Structure Learning of SPNs

In the seminal work of SPNs [6], the structure of SPNs was built in the application-oriented manner. After that, structure learning for SPNs has been introduced in [4, 11]. Dennis & Ventura [11] collected variables using regional relationship to find better structure to solve the image completion task. Gens & Domingos [4] firstly proposed batch-type structure learning method by splitting the variables into mutually independent subsets toward compact representation of joint distribution minimizing the representation power loss.

As a brief introduction, structure learning in [4] is a recursive procedure given dataset T and set of variables V. At first, it checks whether the variables can be split into mutually independent subsets. If possible, the split recursions are done for each subset, and return the products of the resulting SPNs (building internal product nodes). Otherwise, the instances T are clustered into similar subsets, and it returns the weighted sum of the resulting SPNs (building internal sum nodes). Weights for child nodes are determined to be proportional to the number of the assigned instances. At the end of the recursive process, all leaves consist of a univariable node.

At the clustering process, they use a naïve Bayes mixture model to pick most likely component in the mixture, where all variables are independent conditioned on the cluster. They use a hard expectation-maximization (hard EM) algorithm, where each instance is wholly assigned to only its most probable cluster in expectation.

3 Online Incremental Structure Learning Methods

Online structure learning of PGMs is a challenging task because there is no guarantee that new incoming instances follow the learnt model. If new instances are not explained with the structure, it should be modified. Fortunately, in the case of SPNs, online structure learning is deeply related to hard clustering, so we use this property.

We first suggest simple mini-batch incremental clustering problem in Algorithm 1. In Algorithm 1, new instances are assigned to one of the existing clusters. New clusters are added with new instances if they are needed. To find an appropriate number of clusters k, we increase the number of clusters one by one until likelihood does not increase any more than threshold.

```
Algorithm 1. IncrementalClustering(T,V,M)
input: set of mini-batch instances T, set of variables V,
a cluster model M
output: sets of instances assigned to the existing cluster
{T_i}, sets of instances assigned to the new cluster {T_j}
likelihood = -inf
while true
  while model M is converge
    assign T with variables V and model M
    update model M with T
  end
  calculate likelihood
  if increase of likelihood is less than threshold
    break
  end
  add a new cluster to the model M
end
```

We can extend the above clustering process to learn the structure of SPNs. Algorithm 2 illustrates online incremental learning algorithm for SPNs. The algorithm hierarchically adds new child nodes onto the sum nodes in whole layers. The clustering process is used in algorithm 2 as one part. It basically uses the distributions of child nodes. If there is no model for applying to new cluster, however, it also use naïve Bayes model, as the structure learning on the previous study does. After clustering, existing child nodes are augmented recursively, whereas new child nodes are constructed by previous structure learning methods. Our learning method is hybrid of methods in parameter learning [6] and structure learning [4] as illustrated in Figure 2. This method also use hard EM of SPNs as previous methods does. Previous studies show this hierarchical hard clustering strategy is powerful in practice and verify our new learning method is valid.

If the model explains new data well, the structure isn't changed much. Otherwise, the SPN increases their nodes to express new data. The method makes models learn different tree-structure of SPNs if the incoming order of data flow is shuffled with the same data stream.

```
Algorithm 2. AugmentSPN(T,V,M)
input: set of mini-batch instances T, set of variable V,
and an SPN M
output: an augmented cluster model M'
if root of M is univariate node
  M = ParameterLearnNode(T,V,M)
elseif root of M is product node
  for each child node M_k of root
    M_k = AugmentSPN(T,V_k, M_k)
  end
elseif root of M is sum node
  ({T_i},{T_j}) = IncrementalClustering(T,V,M)
  for each instance sets of i-th existing cluster
    M_i = ParameterLearnNode(T_i, V, M_i)
    M_i = AugmentSPN(T_i, V, M_i)
  end
  for each instance sets of j-th new cluster
    M_j = StructureLearnSPN(T_j, V)
  end
end
```

Note that, on the other hands, the variable subsets split by the product node will not change after the product node is generated once. If they are not mutual independent on new data, the product node cannot fully explain them. It may cause the model heavier with meaningless product nodes. It is a major limitation of our method.

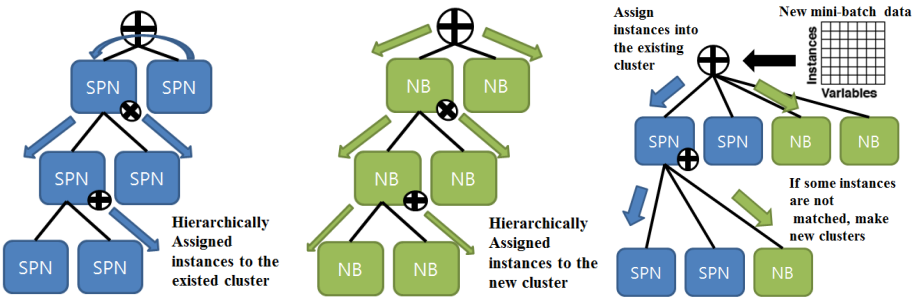


Fig. 2. Parameter learning [6] (left); Structure learning [4] (middle); Online incremental structure learning of SPNs (right). Online structure learning method is hybrid of methods in parameter learning and structure learning. Arrows indicate flows of assigning instances to the node.

4 Experiments

We evaluated online incremental structure learning methods on variants of the "handwritten optic digits" [12] to illustrate our argument. These digit data include an image pixel and a digit class. However, pixels are binarized for our experiments. We make two

settings: "homo" and "hetero." In the hetero setting, we only reorder the dataset according to the class label. In other words, the models first meet all the digit0 images, then the digit1 images, and so on. The homo setting uses only the dataset's own order, and it yields a stable probability distribution of data. The hetero setting, however, yields a dynamically changing probability distribution of data in the mini-batch task.

We evaluated the suggested model in a mini-batch environment. The number of mini-batches was 16. Three models were compared. The first is "classical online learning", which uses only the first mini-batch for structure learning which following previous research studies. The second is "online ensemble learning" which ensembles 16 SPNs constructed by "classical online learning" method. "Classical online learning" methods, It is possible that more complex model may do better, which is why this second model used in the comparison. Our models are quite larger than the first comparable model, but slightly smaller than the second model. The third model is "batch learning," which uses the whole dataset for structure learning and is exactly same as the classical methods used in previous studies.

We tested two methods for evaluating performances, one of which measures likelihood. The other goal is to infer the probability of a subset of the variables (the query) given the values of another (the evidence). We used 50% of the variables as the query and 30% of the variables as the evidence in the experiments.

Figure 3 shows the performance results of the various learning methods as a hetero handwritten dataset arrives. The suggested model not only outperforms naïve online models, but also achieves the performances of batch structure learning. The results imply that the suggested learning method represents well the probability distribution of the data. We also catch that "online ensemble learning" method do better than "classical online learning" methods, which means that previous studies may not have fully tuned their own model.

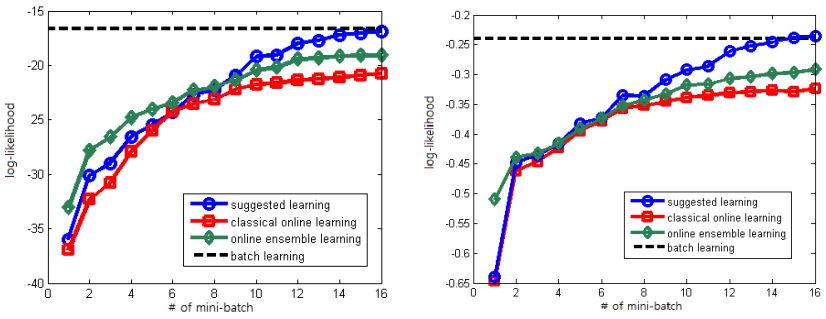


Fig. 3. Log-likelihoods (left). Average conditional log-likelihoods for arbitrary query and evidence (right).

We also investigated the form of the structure of changing SPNs. According to different characteristics of the order in which the data arrive, the structure changes differently. First, the complexities of SPNs are different. Figure 4 shows that, in the hetero setting, the models need more nodes to represent the probability distribution as new data arrive. Second, the structure of SPNs or the numbers of child nodes of a root

differ. Figure 5 shows that the different sequence-order of data evolves different forms of structure. Figure 5 (a, c) shows the results of the first mini-batch step, and Figure 5 (b, d) shows the result of the final mini-batch step. Figure 5 (a-b) shows the structure change of the homo setting when distribution of data is balanced. When data with similar distribution arrive, SPNs increase their depth. Figure 5 (c-d) shows the structure change of the hetero setting when the order of data is according to the class. When data with dynamically changing distribution arrive, SPNs increase their width.

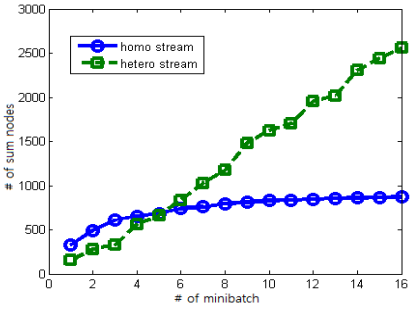
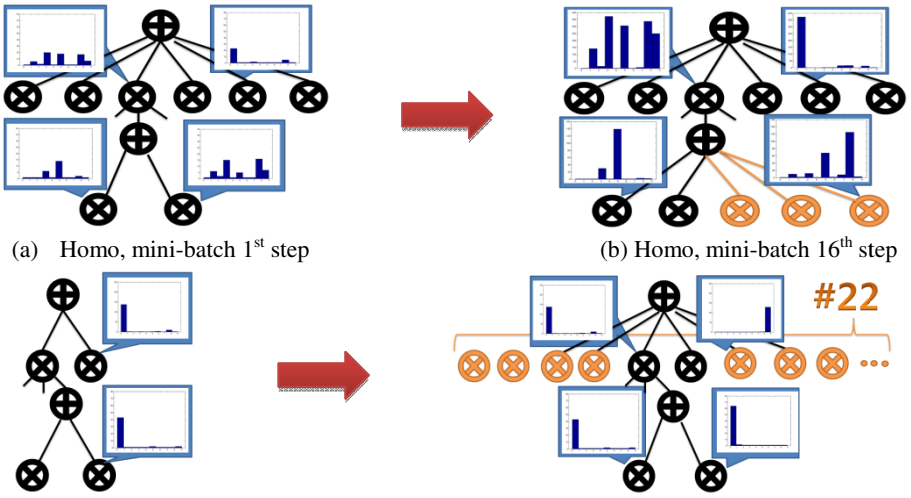


Fig. 4. Growth of complexity. If incoming data are similar to the distribution of the model, the complexity of the model converges. Otherwise, the complexity of the model increases.



(a) Homo, mini-batch 1st step (b) Homo, mini-batch 16th step

(c) Hetero, mini-batch 1st step (d) Hetero, mini-batch 16th step

Fig. 5. Different structures evolve according to different orders of datasets

5 Conclusion

In this paper, an online incremental learning method for SPNs was suggested and their performances were tested. In our method, the number of hidden units and even

layers are evolved dynamically on incoming data to follow the changing distribution of data.

This paper is also a source of motivation for some future studies. First, our methods should be verified on larger-scale data as online learning tasks may be needed in large-scale data environments. Second, the concept of incremental learning can also be used in batch learning with non-parametric Bayesian methods. By applying a non-parametric Bayesian technique, we could more plausibly learn the structure of a Bayesian network. We hope this property can be used for lifelong learning to allow the model to catch new concepts and concept drift [13].

Acknowledgements. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2010-0017734, Videome), supported in part by KEIT grant funded by the Korea government (MKE) (10035348, mLife).

References

1. Chickering, D.M., Heckerman, D., Meek, C.: A Bayesian approach to learning Bayesian networks with local structure. In: Conference on Uncertainty in Artificial Intelligence, vol. 13 (1997)
2. Ravikumar, P., Wainwright, M.J., Lafferty, J.D.: High-dimensional ising model selection using L1-regularized logistic regression. The Annals of Statistics 38(3), 1287–1319 (2010)
3. Bengio, Y.: Learning deep architectures for AI. Foundation and Trends in Machine Learning 2(1), 1–127 (2009)
4. Gens, R., Domingos, P.: Learning the structure of sum-product networks. In: International Conference on Machine Learning, vol. 30 (2013)
5. Zhou, G., Sohn, K., Lee, H.: Online incremental feature learning with denoising autoencoder. In: International Conference on Artificial Intelligence and Statistics, vol. 15 (2012)
6. Poon, H., Domingos, P.: Sum-product networks: A new deep architecture. Uncertainty in Artificial Intelligence 27 (2011)
7. Zhang, B.-T., Muhelnbein, H.: Synthesis of sigma-pi networks by the breeder genetic programming. In: Proceedings of First IEEE Conference on Evolutionary Computation (1994)
8. Darwiche, A.: A differential approach to inference in Bayesian networks. In: Conference on Uncertainty in Artificial Intelligence, vol. 16 (2003)
9. Gens, R., Domingos, P.: Discriminative learning of sum-product networks. In: Advances in Neural Information Processing Systems, vol. 25 (2012)
10. Amer, M.R., Todorovic, S.: Sum-product networks for modeling activities with stochastic structure. In: Computer Vision and Pattern Recognition (2012)
11. Dennis, A., Ventura, D.: Learning the architecture of Sum-Product networks using Clustering on variable. In: Advances in Neural Information Processing Systems, vol. 25 (2012)
12. UCL Machine Learning Depository, `http://archive.ics.uci.edu/ml/`
13. Zhang, B.-T., Ha, J.-W., Kang, M.: Sparse population code models of word learning in concept drift. In: Proceedings of the Annual Meeting of the Cognitive Science Society, vol. 34 (2012)

Kernel Polarization Based on Cooperative Clustering

Weiwei Cao[1], Chuanhuan Yin[1], Shaomin Mu[2], and Shengfeng Tian[1]

[1] School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
{11120427,chyin}@bjtu.edu.cn, msm@sdau.edu.cn
[2] School of Computer and Information Engineering, Shandong Agricultural University, China
sftian@bjtu.edu.cn

Abstract. In recent years, kernel methods are used in many applications, such as text classification and gene recognition. The parameters of kernels are empirically decided by the context of application. In order to select the appropriate kernel parameters, kernel polarization is presented as a universal kernel optimality criterion, which is independent of the classifier to be used. However, kernel polarization has several disadvantages, leading to the inconvenience of applying such method. In this paper, a clustering algorithm called Cooperative Clustering is integrated with kernel polarization. The experimental results showed the effectiveness of the approach.

Keywords: Support vector machine, Kernel polarization, Cooperative clustering.

1 Introduction

Kernel methods [1, 2] have been used in many applications of classification, such as text classification, intrusion detection, and so on. The idea behind kernel methods is that it can map the data points into a high dimensional feature space by introducing kernel functions. Therefore, the classification accuracy is related to mapping functions, i.e. the kernels. In order to achieve higher performance, various kernels are designed, including Radial Basis Function (RBF) kernels [3], string kernels and tree kernels [2]. Besides, the parameters of kernels [4] are closely related to the classification accuracy. In short, kernels and their parameters are critical factors in classification problems.

Kernel polarization (KP) [5] is proposed as a criterion in measuring whether the kernel function is good especially for the parameters selection in the model. However, the KP has its disadvantages at the same time. When the size of the two kinds of samples is biasing, the larger one will have great influence on the result which would make a large error. Besides, the KP only focuses on the overall optimization of the two kinds of samples and ignores the boundary distribution which will also lead to a large error of the result. Additionally, it needs all the samples to be trained which would make the efficiency very low.

According to the above problems, we have made some improvements such as the improved kernel polarization (IKP), improved kernel polarization based on clustering (CIKP). But the methods proposed above only improved on a particular aspect of the defect. In this paper, we propose a method called "improved kernel polarization based

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 228–234, 2013.

on cooperative clustering (CCIKP)" for parameter selection. On one hand, it highlights the influence of the boundary sample; on the other hand, it greatly reduces the numbers of samples which improves the learning efficiency. Finally, we test our method with RBF kernels [6, 7] and make a comparison with the other three methods.

The rest of this article is organized as follows. In section 2, we give a short description of KP. In section 3, IKP is introduced. In section 4, we show a method of CIKP. And our method CCIKP is presented in detail in section 5. Experimental results showing its effectiveness are presented in section 6 and at last we draw conclusions with some future works in section 7.

2 KP

Given a set of labeled pointswhere, the kernel function defines a symmetric positive definite kernel matrix K by $K_{ij} = k(x_i, x_j)$. Let $y = (y_1, \ldots, y_n)^{\mathrm{T}}$, the ideal kernel matrix would be yy^{T} which perfectly suits the training data. Then the KP is defined as follows [5]:

$$\varphi = < K, yy^{\mathrm{T}} > = \sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j K(x_i, x_j) = \sum_{y_i = y_j} k(x_i, x_j) - \sum_{y_i \neq y_j} k(x_i, x_j) \tag{1}$$

Through the above formula, it is easy to find that the KP measures the similarity of the kernel with yy^{T} on the training data. And obviously, φ will increase if the similarity represented by the kernel turns larger for input patterns of the same class while that for patterns of different classes turns smaller. This is the intuitive idea behind preferring a kernel with high kernel polarization in the machine learning. However, this method has some disadvantages, which are as follows:

(1) When the size of the two kinds of samples is biasing, the larger one will have great influence on the result which would make a large error.

(2) The KP only focuses on the overall optimization of the two samples but ignores the boundary distribution which will also lead to a large error of the result.

(3) This method needs all the samples to be trained which reduce the efficiency.

3 IKP

As we know, when there is a big difference between the numbers of the two samples, the performance of KP will be poor. So people have made some efforts in order to surmount this problem. For the purpose of eliminating the unbalance of the two samples' numbers, they introduce a new variable r_i and call this method IKP:

$$J = \sum_{i=1}^{n}\sum_{j=1}^{n} r_i r_j K(x_i, x_j) \quad \text{where} \quad r_i = \begin{cases} 1 & y_i = 1 \\ -n_0/n_1 & y_i = -1 \end{cases} \tag{2}$$

where n_0 is the positive samples' number and n_1 is the negative samples' number.

This method takes the number of two kinds of samples into consideration, and makes great contributions in eliminating the unbalance of the two samples' numbers.

4 CIKP

In the method of IKP, all samples are used in the training process, which inevitably requires longer training time. In order to reduce the time complexity, the idea of clustering can be used to preprocess the training samples. We call this method CIKP.

In CIKP, each class can be divided into several clusters, and each cluster can be represented by its central sample. For binary classification, the positive class is divided into p clusters and the negative one is divided into q clusters. Then every cluster is replaced with its cluster center to as the training sample of IKP, and the sample number of each cluster is used as the weight of its cluster center. For multiple class classification, there are some differences. Suppose that there are r classes, and each class has p cluster centers. For the *i-th* classifier, the class i is set with + and divided into p clusters, the other classes are set with – and divided into q clusters, where $q=p(r-1)$. It is similar to binary classification. If the size of the two kinds of samples is biasing, we can set $q>p$, and then the similar effect can be achieved, too.

Suppose the number of the training set in class i is n_0, and that in other classes is n_1, and $Z=\{(z_1,y_1),(z_2,y_2),\ldots,(z_n,y_n)\}$ is the sample set of KP, and t_i is the weight of the sample Z_i. Then the CIKP is defined as follows:

$$J=\sum_{i=1}^{n}\sum_{j=1}^{n}s_i s_j K\left(z_i, z_j\right) \quad \text{where} \quad s_i=\begin{cases} t_i & y_i=1 \\ -t_i n_0/n_1 & y_i=-1 \end{cases} \tag{3}$$

5 CCIKP

As described above, the method of CIKP could reduce the number of the training set, but it does not consider the boundary samples. However, those ones often have a great influence on the result. Therefore, cooperative clustering is integrated into IKP, called CCIKP. And CCIKP can highlight the influence of the boundary samples.

5.1 Method of Cooperative Clustering

The thought of cooperative clustering is put forward by Tian et al [8]. Suppose that there are two-class data sets X^+ and X^-. With k-means [9] algorithm, the two data sets are divided into p clusters and q clusters where $p \le q$. Now choose the cluster centers of class + and class – marked as $v^+=\{v_1^+,\ldots,v_p^+\}$ and $v^-=\{v_1^-,\ldots,v_q^-\}$. H represents the distance matrix between the two sets v^+ and v^-. The *ij-th* entry h_{ij} of matrix H can be computed as

$$h_{ij}=\left\|v_i^+-v_j^-\right\|^2 \quad i=1,...,p,\ j=1,...,q \tag{4}$$

We take out each pair of cluster centers with smallest distance from sets v^+ and v^- iteratively according to matrix H. The two cluster centers in $\left(v_a^+,v_b^-\right)$ should move towards each other. Let r_a^+ be the average radius of cluster *a* in class + and r_b^- be the average radius of cluster *b* in class -. So we have

$$r_a^+=\frac{1}{n_a^+}\sum_{x\in X_a^+}\left\|x-v_a^+\right\| \qquad r_b^-=\frac{1}{n_b^-}\sum_{x\in X_b^-}\left\|x-v_b^-\right\| \tag{5}$$

where n_a^+ and n_b^- are sample numbers in clusters X_a^+ and X_b^- respectively. Then each pair $\left(v_a^+,v_b^-\right)$ is updated as follows

$$v_a^+=v_a^++\lambda\frac{r_a^+}{r_a^++r_b^-}\left(v_b^--v_a^+\right) \qquad v_b^-=v_b^-+\lambda\frac{r_b^-}{r_a^++r_b^-}\left(v_a^+-v_b^-\right) \tag{6}$$

where $\lambda\in(0,1)$ is the quantity who controls the distance between v_a^+ and v_b^-.

The whole procedure of the cooperative clustering is as follows:

(1) Partition X^+ and X^- into *p* and *q* clusters with *k*-means respectively, and get the cluster centers $v^+=\left\{v_1^+,\ldots,v_p^+\right\}$ in class + and $v^-=\left\{v_1^-,\ldots,v_q^-\right\}$ in class -.

(2) Set $V^s=\{\}$, compute the matrix $H=\left\{h_{ij}\right\}\ 1\le i\le p,1\le j\le q$ with eq (4).

(3) Find the smallest element in H, update the center pair $\left(v_a^+,v_b^-\right)$ with eq (6).

(4) Add $\left(v_a^+,v_b^-\right)$ into V^s and delete the row *a* and the column *b* from matrix H.

(5) If the number of pairs in set V^s is less than *p*, go to step (3).

(6) Partition X^+ and X^- into clusters $\left\{X_1^+,X_2^+,\ldots,X_p^+\right\}$ and $\left\{X_1^-,X_2^-,\ldots X_q^-\right\}$ with updated cluster centers, and compute new cluster centers in class + and in class -.

(7) If the partition is changed in step (6), go to step (2).

With the above procedure, we can find *p* pairs of cluster centers in V^s. Each pair crosses the boundary of the two classes.

5.2 CCIKP

Cooperative clustering can generate *p* pairs of clusters, and each pair crosses the boundary of the two classes. After cooperative clustering, the *p* pairs of clusters are as the samples of IKP, and the weight of each cluster is 1. At the same time, the cluster centers of the other samples except the *p* pairs are as the samples of IKP, too, but the weight of each cluster center is the number of its samples. Therefore, the sample set

represents all the samples. It highlights the influence of the samples near to the boundary while keeping the small quantity of the training set.

Suppose the number of the training set in class i is n_0, and that in other classes is n_1, $Z=\{(z_1,y_1),(z_2,y_2),\ldots,(z_n,y_n)\}$ is the sample set of the IKP, and t_i is the weight of sample z_i except the p pairs of clusters. Let V^s represents the p pairs of clusters which around the boundary of the two classes. Then the CCIKP is defined as follows:

$$J=\sum_{i=1}^{n}\sum_{j=1}^{n} s_i s_j K(z_i, z_j) \quad \text{where} \quad s_i=\begin{cases} t_i & y_i=1 \cap y_i \notin V^s \\ -t_i n_0/n_1 & y_i=-1 \cap y_i \notin V^s \\ 1 & y_i \in V^s \end{cases} \tag{7}$$

CCIKP is the improvement made on the basis of foregoing methods. It overcomes the drawbacks of KP and takes the boundary samples into consideration.

6 Experiments

Our experiments are based on RBF Kernels, which is defined as follows [3]:

$$K(x,x') = \exp[-b\|x-x'\|^2] \tag{8}$$

We choose kernel polarization as the criterion for kernel selection [10] and use the gradient-based method [11] for learning its width parameter b. So we have

$$\frac{\partial J}{\partial b} = -\sum_{i=1}^{m}\sum_{j=1}^{m} s_i s_j \|z_i - z_j\|^2 \exp\left[-b\|z_i - z_j\|^2\right] \tag{9}$$

The following is the algorithm of the gradient-based method and the parameter of the step length marked as *stp* is self-driven. And the original value of b is set to 1.

for $p = 0$ to 100

{ $jg = \partial J/\partial b$; $st = jg * stp$; if $(p == 0 \,\|\, st > 0.1)\{stp = 0.1 / \text{fabs}(jg); st = 0.1;\}$ $b \mathrel{+}= st;$}

To test the effectiveness of the above methods, we apply them to four benchmark data sets from the UCI Machine Learning Repository [12] summarized in Table 1.

Table 1. Data sets used in the experiments

Problem	#training data	#testing data	#attribute	#class
pima	537	231	8	2
vehicle	592	254	18	4
segment	1617	693	19	7
pendigitst	7494	3498	16	10

Among many learning procedures for Support Vector Machine (SVM), we choose Platt's sequential minimal optimization (SMO) [13] as training algorithm. The parameter λ of the cooperative clustering algorithm is fixed to $\lambda=0.3$. The parameter C in SMO is chosen the best one among {0.1, 1, 10, 100} according to the testing accuracy. The number of clusters of each class is set to about 10% of the training data. For problems pima, vehicle and segment, we randomly choose 70% for training and test the remaining 30%. Then we run it for 10 times and choose the average values as the results. For problem pendigitst, training set and testing set are provided separately in the UCI repository. For all problems, we linearly scale all training data to be in [-1,1] for features with plus and minus values and in [0,1] for features with only plus values. Then testing data are scaled accordingly. The results are shown in Table 2 to Table 5.

Table 2. Classification results on the pima data set for different methods

Method	#Training data for *b*	C	Training acc(%)	Training time(s)	Testing acc(%)	Testing time(s)
KP	537	10	77.28	2.52	74.03	0.01
IKP	537	1	81.56	2.50	76.62	0.01
CIKP	52	1	81.94	0.23	75.76	0.01
CCIKP	537	1	81.56	2.56	76.62	0.01

Table 3. Classification results on the vehicle data set for different methods

Method	#Training data for *b*	C	Training acc(%)	Training time(s)	Testing acc(%)	Testing time(s)
KP	592	10	23.14	12.10	24.80	0.05
IKP	592	10	88.85	15.10	76.77	0.05
CIKP	56	10	90.20	2.81	76.38	0.05
CCIKP	331	10	89.86	6.87	76.77	0.05

Table 4. Classification results on the segment data set for different methods

Method	#Training data for *b*	C	Training acc(%)	Training time(s)	Testing acc(%)	Testing time(s)
KP	1617	100	14.16	159.29	14.72	0.72
IKP	1617	100	98.21	179.69	96.54	0.75
CIKP	161	100	98.27	11.86	96.68	0.76
CCIKP	555	100	98.21	33.25	96.54	0.75

Table 5. Classification results on the pendigitst data set for different methods

Method	#Training data for *b*	C	Training acc(%)	Training time(s)	Testing acc(%)	Testing time(s)
KP	7494	10	9.59	4812.94	9.61	24.11
IKP	7494	10	99.91	4923.42	98.71	24.23
CIKP	740	10	99.96	126.88	98.66	24.24
CCIKP	1918	10	99.91	462.14	98.71	24.19

7 Conclusion

In this paper, we introduced a new kernel optimality criterion, which is called CCIKP. It overcomes the problem that the traditional kernel polarization is sensitive to the disequilibrium of the number of samples. It not only considers the overall sample, but also takes the boundary samples into account. Moreover, it greatly reduces the training time while keeping the classification accuracy. In the experiments, the performance is demonstrated with some UCI machine learning benchmark examples. And the experimental results show that our method is very effective. Future investigation will focus on the selection of the sample centers, decreasing their redundancy in CCIKP.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (No. 61105056), the Fundamental Research Funds for the Central Universities, Shandong Provincial Natural Science Foundation, China (No. ZR2012FM024).

References

1. Müller, K., Mika, S., Rätsch, G., Tsuda, K., Schölkopf, B.: An introduction to kernel-based learning algorithms. IEEE Trans. on Neural Networks 12(2), 181–202 (2001)
2. Taylor, J.S., Cristianini, N.: Kernel methods for pattern analysis. Cambridge University Press (2004)
3. Buhmann, M.D.: Radial basis functions, pp. 1–38. Cambridge University Press, Cambridge (2000)
4. Cristianini, N., Campbell, C., Shawe-Taylor, J.: Dynamically adapting kernels in support vector machines. In: Advances in Neural Information Processing Systems (NIPS), vol. 11 (1998)
5. Baram, Y.: Learning by kernel polarization. Neural Computation 17, 1264–1275 (2005)
6. Chung, K.M., Kao, W.C., Sun, C.L., Wang, L.L., Lin, C.J.: Radius margin bounds for support vector machines with RBF kernel. Neural Computation 15(11), 2643–2681 (2003)
7. Chang, Q., Wang, X.-L., Lin, Y.-M., Wang, X.-Z., Yeung, D.S.: Support vector classification and Gaussian kernel with multiple widths. Acta Electronica Sinica 35(3), 484–487 (2007)
8. Tian, S., Mu, S., Yin, C.: Cooperative Clustering for Training SVMs. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 962–967. Springer, Heidelberg (2006)
9. Huang, Z.: Extensions to the *k*-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2, 283–304 (1998)
10. Wang, T.-H., Huang, H.-K., Tian, S.-F.: Learning General Gaussian Kernels by Optimizing Kernel Polarization. Chinese Journal of Electronics 18(2), 265–269 (2009)
11. Glasmachers, T., Igel, C.: Gradient-based adaptation of general Gaussian kernels. Neural Computation 17(10), 2099–2105 (2005)
12. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases. Department of Information and Computer Science. University of California, Irvine (1998)
13. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: Advances in Kernel Methods: Support Vector Learning, pp. 185–208. MIT Press, Cambridge (1999)

Estimation Methods of Presumed Income

Valter E. Silva Júnior[1,⋆], Renata M.C.R. Souza[1],
Getúlio J.A. Amaral[2], and Hélio G. Souza Júnior[3]

[1] Centro de Informática (CIn), Universidade Federal Pernambuco, Recife, Brazil
[2] Departamento de Estatística, Universidade Federal Pernambuco, Recife, Brazil
[3] Departamento de Estatística Aplicada e Biometria, Universidade Federal Rural Pernambuco Recife, Brazil
valteresj2@gmail.com,
rmcs@cin.ufpe.br,
gjaa@de.ufpe.br,
helio.souza@ufrpe.br

Abstract. This paper will be presented regression models to estimate the assumed income that is of utmost importance to the credit market since the client does not prove your income. The proposed models are lognormal and gamma which is a generalized linear model, both will be compared who perform better will be chosen, where the result of the chosen model will undergo an addition of a noise to get a better estimated. Every analysis and simulation were implemented in [7].

Keywords: Income Presumed, lognormal, Generalized Linear Model, Kolmogorov-Smirnov, Holdout.

1 Introduction

One of the most frequent problems that companies face when making decisions is the vulnerability in relation to the accuracy of information of monthly income that the consumer spends to fill a business proposal. It is not easy to confirm this income, the risk of adulteration of documents and the difficulty of getting confirmations with employers and providers.

It is a fact that asymmetric information and high transaction costs in the pursuit of income data in specialized bureaus are the main problems and can cause inefficiency in the market for the determination of income. Usually the individual has more information about yourself (both positive and negative) of the company and avoid rising defaults.

It is natural to expect that there is an accumulation of cases of bad customers greater the commitment of monthly income. The larger the monthly commitment in relation to presumed income, the higher the default. To avoid these situations, companies have used the model of presumed income, which enables the intelligent use of information. [1] start from an assumption of lognormal distribution

⋆ The authors world like to thank CNPq, CAPES and FACEPE (Brazilian Agencies for their financial support.)

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 235–241, 2013.

of income. In this article we will use the lognormal model against gamma generalized linear model. The data to be used in our project is from the National Survey by Households Sample (2009), witch was conducted by the Brazilian Institute of Geography and Statistics. The data provides a comprehensive set of information about the population, in its many aspects. The publication also includes technical notes on which stand the concepts and definitions used in research, methodological considerations that allow to know the main aspects of its historical evolution, and the sampling plan.

In addition to this information, are listed at the end of the publication, all the topics investigated by the survey since 1992, for the following characteristics: household, general data of residents, migration, education, child labour 5-9 years of age, work, and fertility.We are going to just pick some variables from this huge data set to do our statistical analysis under the proposed model framework.

2 Model Specification

Our goal is to create a model for an estimated income very close to the real income by gender, education level, occupation, age and federative unit, where these covariates have a significant influence on income. Thus, in our model, we use the income variable response that can be treated as a continuous random variable. CHAID technique was used to classify occupations see [6], Figure 1 shows a piece of this classification.

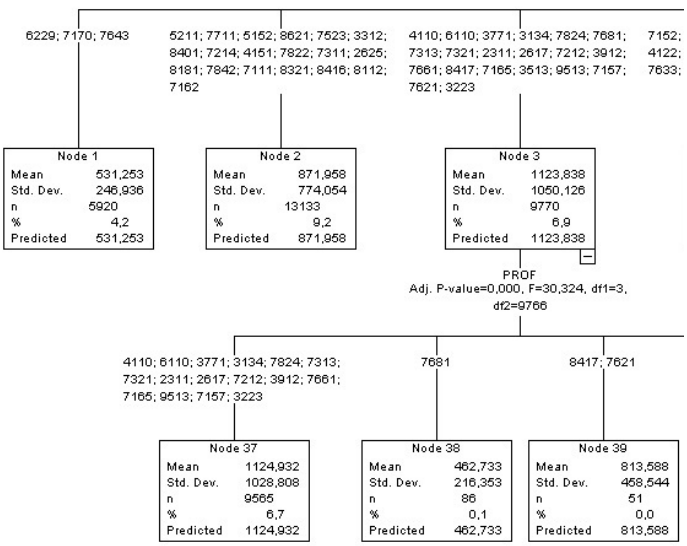


Fig. 1. Classification of occupations

The predictor variables included are:

Gender: Male/Female
Education: Primary/Elementary, Middle, Top completed, Master's degree/PhD and Literacy
Age: be between 18-65
Federative: Unit are the 27 states including the Federal District
Profession: Included are 125 groups with 599 professions

The age can be treated as a continuous variable. All other predictor variables are categorical, dummy variables are used in the definition model. Another issue is that after checking the data, it is found that income has a decreasing exponential. So it will be two models are examined them, lognormal and gamma generalized linear model with logarithmic link function. Below is the structure of the mathematical model of income with their covariates,

$$Income = 1 + Gender + Education + Age + Federative + Profession \quad (1)$$

Because an overall intercept is included in the model, the number of dummy variables within each predictor should be one fewer than the level of that predictor. For instance, there are 5 levels of education as described before, so there should be 4 dummy variables associated with the predictor education.

3 Model Training

This section presents the application of the proposed models in the training base, the method Holdout is to divide the whole data into two mutually exclusive subsets, one for training (estimation of parameters) and the other for testing (validation), see [5]. The data set can be split in equal amounts or not. The proportion used for dividing the base was 75% for training and 25% for the test.

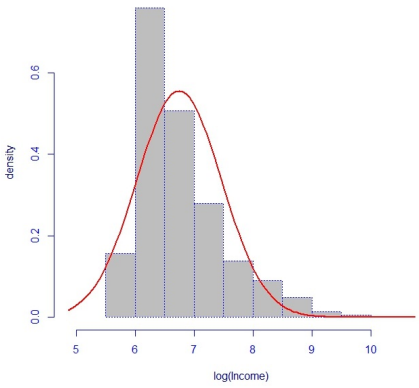


Fig. 2. Graph of model fit lognormal

Figure 2 illustrates the setting of the dependent variable transformed with the log function, where the exponential function is used which is the inverse of the log to obtain the estimated income, the red line represents the adjustment of the model.

The red line represents the estimate of the lognormal model shows that the model fits well the income.

Figure 3 illustrates the model fit with gamma log link function, where the red line approximates well to the training data.

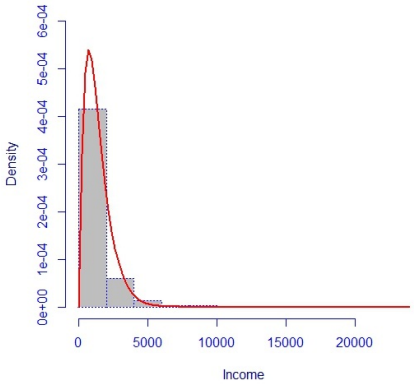


Fig. 3. Graph of model fit gamma

Then the test is performed adequacy to the two models, for the lognormal model was used F test (see [4]) based on the hypothesis (3) and the gamma model is used the Wald statistic (see [2]) following the hypothesis (2) and (3), the level of significance was set at $\alpha = 5\%$.

$$\begin{aligned} H_0 &= \beta_1 = \beta_2 = ... = 0 \\ H_1 &= \beta_1 \neq \beta_2 \neq ... \neq 0 \end{aligned} \tag{2}$$

$$\begin{aligned} H_0 &= The\ model\ is\ suitable \\ H_1 &= Otherwise \end{aligned} \tag{3}$$

therefore, we have the p-value of the lognormal model is $2.2e^{-16}\%$ which is a value lower than the significance level α, ie, we reject the null hypothesis (2), where the alternative hypothesis states that the lognormal model is suitable for the F test, while the Wald test in the gamma model had a p-value of 56% which is a value greater than α, so do nor reject the null hypothesis (3) soon the gamma model is also suitable. The choice of model will be based from the $pseud - R^2$ model that present the greatest value will be used to analyse the test base. The $pseud - R^2$ is denoted by the following:

$$pseud - R^2 = (correlation(value\ Predict, value\ True))^2 \tag{4}$$

the lognormal model obtained the $pseud - R^2$ of 0.66, while the gamma model had a value of 0.72, so the model chosen is the gamma.

Table 1 shows the approach of estimated income compared to income from real quantiles ranging from $1\%, 25\%, 50\%, 75\%$ *and* 99%.

Table 1. Income estimated versus actual income training

	1%	25%	50%	75%	99%
True	510	700	900	1470	8000
Predicted	551	767	924	1412	7782

Table 2 shows the test results obtained on the basis based on the training model estimated gamma follows the comparison of the quantile previously determined between the estimated income versus the true.

Table 2. Income estimated versus actual income test

	1%	25%	50%	75%	99%
True	510	700	900	1481	8000
Predicted	565	764	945	1401	7338

From the estimated value is calculated by the equation (4), the value obtained was 0.712, which means 71.2% of the dependent variable can be explained by the covariates in the model.

4 Simulation and Results

Since the problems of estimating income, this section will be presented the technique for reducing the bias of the model. Regression models always show bias, ie, information not captured by the model, it is simulated with the residues randomly with a mean and standard deviation obtained in model training.

The residue model training follows a normal distribution with a mean $2.12e-11$ and standard deviation 0.26, Figure 2 illustrates a simulation done with a random generator with the mean and standard deviation of residue obtained from modelling, where the red line represents the estimation density residue and the red line represents the values of the generation of random numbers normal, it is apparent that the red line and blue are very close.

Then, the test was performed adhesion of the residue model training from the residue generated randomly the average and standard deviation of the model residue. Test was used to Kolmogorov-Smirnov (see [3]), the p-value of the test will not be used because of the instability with large samples, so the value of the statistic obtained was 0.0012, ie, the closest to zero is greater the approach of randomly generated residue to the residue of the model.

The simulation proposed in this paper will be the addition of waste generated randomly with the distribution and the parameters obtained by the analysis of the residue of the training model with linear predictor with logarithmic link

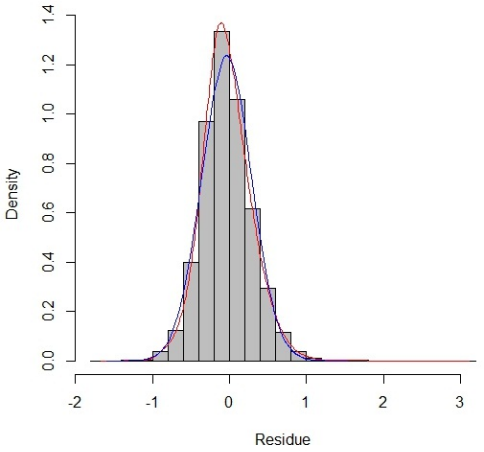


Fig. 4. Approximate model of the residue by residue generated randomly

function to get the original value is applied to the inverse logarithmic function that is exponentially, as shown in equation (5).

$$\eta = \exp\left(\hat{\beta}X + N(2.12e^{-11}, 0.26)\right) \tag{5}$$

From this new linear predictor was performed a simulation of size 3000 and after the simulation was obtained average quantiles and pseudo-R^2. Table 3 shows the results of quantile

Table 3. Income estimated versus real income simulation

	1%	25%	50%	75%	99%
True	510	700	900	1470	8000
Predicted	513	706	912	1472	7985

Therefore, correcting the bias of estimated income base test was closest real income and had a significant increase in the $pseudo - R^2$ from 0.712 to 0.783. Therefore, the model with gamma correction in vies presents a better result than without correction. Below is presented the scope of the algorithm:

1. Select a random sample of 75% j the index $i = 1, 2, ..., N$, where N is the size of the base.
2. With basic training $X_{[j,p]}$, run the generalized linear model with gamma their p covariates.
3. Then get the β and the estimated residue model ϵ.
4. Mount the base test $X_{[-j,p]}$.
5. Determine the size of the simulation $n = 3000$.

(a) Generate a looping from 1 to n.
(b) Generate random numbers R with mean and standard deviation of residue model.
(c) Calculate the estimated value of the base teste $exp^{(\beta X_{[-j,p]}+R)}$.
(d) Calculate and store the vector of the pseudo-R^2 and the quantiles of the estimated value at each iteration.

6. Return the average pseudo-R^2 and quantiles.

5 Conclusion

The proposed models work well with positive data decrease exponentially, where the gamma model outperformed lognormal, for the model with gamma correction ensures greater accuracy vies assumed income estimated in relation to the actual income from this improvement, the models used in the financial market may possibly have a better score for the client company. The variables predictive played a very important role to estimate the assumed income, variable profession that suffered a supervised classification aiming to reduce the amount of levels which was quite significant for modelling, representing a gain of 32% in the pseudo-R^2 training model. Note that whenever possible to confirm that the information passed to the customer is true that the model can accurately estimate the presumed income of the individual.

Future work includes other features such as the breed of individual who would be an important factor to distinguish better the assumed income and applying the multinomial model to determine which range of income the individual fits and so applying this variable as a new input variable to model chosen.

References

1. Balintfy, J.L., Goodman, S.S.: Socio-Economic Factors in Income Inequality: A Log-Normal Hypothesis. Zeitschrift für Nationallökonomie, pp. 389–402. Springer (1973)
2. Buse, A.: The likelihood ratio, Wald and Lagrange multiplier tests: an expository note. The American Statistician 36, 153–157 (1982)
3. Conover, W.J.: Pratical nonparametric statistics, 3rd edn., pp. 428–433. Moopark, John Wiley & Sons, Inc., New York(1999)
4. Seber, G.A.F., Lee, A.J.: Linear Regression Analysis, 2nd edn., pp. 99–100. Wiley (2003)
5. Kohavi, R.: A study of croos validation and boostrap for accuracy estimation and model selection. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 1137–1143. Morgan Kaufmann Publishers (1995)
6. Ripley, B.D.: Pattern Recognition and neural networks, 1st edn. Cambridge University Press, Cambrigde (1996)
7. R Development Core Team, `http://www.r-project.org/`

Online Model-Free RLSPI Algorithm for Nonlinear Discrete-Time Non-affine Systems*

Yuanheng Zhu and Dongbin Zhao

State Key Laboratory of Management and Control for Complex Systems,
Institute of Automation Chinese Academy of Sciences, Beijing, China
zyh7716155@163.com, dongbin.zhao@gmail.com

Abstract. Policy iteration, as one kind of reinforcement learning methods is applied here to solve the optimal problem of nonlinear discrete-time non-affine system with continuous-state and continuous-action space. By applying action-value function or Q function, the implementation of policy iteration avoids the dependence on system dynamics. Online model-free recursive least-squares policy iteration (RLSPI) algorithm is proposed with continuous policy approximation. It is the first attempt to develop online LSPI algorithm for nonlinear discrete-time non-affine systems with continuous policy. A nonlinear discrete-time system is simulated to verify the efficiency of our algorithm.

Keywords: online policy iteration, Q function, nonlinear discrete-time non-affine system, linear parametrization, RLSPI.

1 Introduction

Reinforcement learning (RL) refers to one kind of methods that try to find optimal or near-optimal policies for complicated systems or agents [1-3]. Policy iteration (PI) is one powerful instrument of RL to solve optimal problems. PI [4] includes two steps: policy evaluation and policy improvement. With iteration of these two steps, PI improves policy constantly and finally achieves the optimal one [5].

As RL developed, function approximation (FA) technique was introduced to RL to solve continuous optimal problems and promoted the development of RL. Such as approximate SARSA [6], TD(λ) [7] and so on. Especially recent years, a new branch of RL, adaptive dynamic programming (ADP) was proposed [2]. Some overviews about ADP are given in [2], [3], [8] and [9].

As FA technique is used for the approximation of value function, parameters of approximation have to be learned based on data. And a lot of online algorithms have developed. SARSA is an online algorithm which modifies value function based on temporal difference (TD) error with gradient method. Si and Wang [10] applied ADP for online learning of under-actuated control systems and presented great

* This work was supported in part by National Natural Science Foundation of China (Nos. 61273136, and 61034002,), and Beijing Natural Science Foundation No. 4122083.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 242–249, 2013.

performance. However, those algorithms make a limited use of data, which does not benefit their application.

To solve those problems, Busoniu *et al.* [11] extended offline least-squares policy iteration (LSPI) [12] to an online LSPI algorithm. And this algorithm employed Q function featured as model-free and the results revealed great performance for online learning. However, a batch least-squares method was used and only discrete-action policies were applied.

In this paper, we focus on a brand new research field of online model-free recursive learning with continuous-action policy approximation for nonlinear discrete-time non-affine systems. An online model-free RLSPI algorithm is proposed using linear function approximation for continuous state and action systems. To the limit of our knowledge, it is the first attempt to combine continuous policy approximation with LSPI for online learning.

This paper is organized as follows. In Section 2, PI method using Q function is introduced to solve optimal control problem of nonlinear discrete-time non-affine system. Then an online model-free algorithm, RLSPI, is proposed in Section 3 to solve this kind of problems online. And a nonlinear example is simulated with the new algorithm. In the end, we have our conclusion.

2 PI for Nonlinear Discrete-Time Non-affine Optimal Problem

2.1 Nonlinear Discrete-Time Non-affine Optimal Problem

In this paper, the nonlinear discrete-time non-affine system is denoted by $x_{k+1} = f(x_k, u_k)$, where $x_k, x_{k+1} \in R^n$, $u_k \in R^m$, $f : R^n \times R^m \to R^n$, and k is the step index. Assume the system is controllable on a compact set Ω and 0 is the equilibrium point. Suppose a negative definite function $r(x_k, u_k)$ is used as the reward at each step, and $h : R^n \to R^m$ represents a policy.

Given a policy h, the following definition specifies its action-value function, or Q function

$$Q^h(x_k, u_k) = r(x_k, u_k) + Q^h(f(x_k, u_k), h(f(x_k, u_k))) . \tag{1}$$

Q^h are negative definite. And the optimal control problem is to find the maximum Q function and optimal policy, namely

$$Q^*(x_k, u_k) = \max_h Q^h(x_k, u_k) , \tag{2}$$

$$h^*(x_k) = \arg\max_u Q^*(x_k, u) . \tag{3}$$

It is important to note that the Q function defined here is undiscounted. So a definition is introduced from [13] to guarantee the validity

Definition 1. (Admissible Policy) A control policy h is defined to be admissible, denoted by $h \in \psi(\Omega)$, if h is continuous on Ω, $h(0)=0$, h stabilizes the system on Ω and for $\forall x_k \in \Omega$, $Q^h(x_k,u_k)$ is finite.

2.2 Policy Iteration

Based on Q function, policy evaluation and policy improvement of PI is presented as follows

PI method

1. Policy evaluation: given an admissible control policy $h^{(i)} \in \psi(\Omega)$, calculate relevant Q function by

$$Q^{(i)}(x_k,u_k) = r(x_k,u_k) + Q^{(i)}(f(x_k,u_k),h^{(i)}(f(x_k,u_k))),\ Q^{(i)}(0,0)=0. \tag{4}$$

2. Policy improvement: generate a new improved policy $h^{(i)}$ using

$$h^{(i+1)}(x_k) = \arg\max_u Q^{(i)}(x_k,u). \tag{5}$$

As the policy is improved over and over again, the optimal policy can be finally obtained.

3 An On-Line Model-Free RLSPI Algorithm

LSPI method was first proposed by Lagoudakis and Parr [12] to utilize the linear property of PI method and apply least-squares method for finite state and action sets. Then Busoniu *et al.* [11] extended this method to an online version for infinite and continuous-state space and finite-action sets. However, their algorithm is not suitable for more general continuous action systems.

Different from offline or online LSPI algorithm using batch least-squares at the end of iterations, a new online model-free RLSPI algorithm is proposed which applies RLS method at each step and uses continuous policy approximation for continuous state and action control problems.

3.1 Q Function and Policy Approximation

As state and action spaces are continuous, approximation of Q function and policy is necessary. Here, linear parametrization technique [13] is used.

A linear parametrization of Q function can be expressed by $\hat{Q}(x,u) = \phi^T(x,u)\theta$, where $\phi(x,u) = [\phi_1(x,u),...,\phi_N(x,u)]^T$ is a vector of N basis functions (BFs), and $\theta \in R^N$ is a parameter vector. Like many others works [13], polynomials are adopted

here as BFs. Suppose $x=[x_1,...,x_n]^T$ and $u=[u_1,...,u_m]^T$ and let polynomial BF ϕ_i have the form $\phi_i(x,u)=x_1^{\alpha_1^i}x_2^{\alpha_2^i}\cdots x_n^{\alpha_n^i}u_1^{\beta_1^i}u_2^{\beta_2^i}\cdots u_m^{\beta_m^i}$.

Similarly, denote the linear parametrization of policy by $\hat{h}(x)=\omega^T\varphi(x)$, where $\varphi(x)=[\varphi_1(x),...,\varphi_M(x)]^T$ and $\omega=[\omega_1,...,\omega_m]\in R^{M\times m}$, while each vector $\omega_j\in R^M$ is associated with action u_j. And the polynomial BF φ_j is defined as $\varphi_j(x)=x_1^{\gamma_1^j}x_2^{\gamma_2^j}\cdots x_n^{\gamma_n^j}$.

3.2 Policy Evaluation with Q Function Approximation

With Q function approximation and policy evaluation, the calculation of parameters vector θ can be implemented using online data. Suppose current policy is $\hat{h}^{(i)}$ and try to solve parameters vector $\theta^{(i)}$. $\{(x_t,u_t,x_{t+1})\}$ denotes online data.

For each sample (x_t,u_t,x_{t+1}), combine $\hat{Q}^{(i)}$ and (4) and we have

$$[\phi(x_t,u_t)-\phi(x_{t+1},\hat{h}^{(i)}(x_{t+1}))]^T\theta^{(i)}=r(x_t,u_t). \tag{6}$$

It is obvious that (6) is a linear to $\theta^{(i)}$. Besides, online samples are collected step by step. So RLS method is applied for learning $\theta^{(i)}$. The whole learning process is presented by

$$\begin{aligned}
&z(t)=r(x_t,u_t)\\
&s(t)=\phi(x_t,u_t)-\phi(x_{t+1},\hat{h}^{(i)}(x_{t+1}))\\
&q(t)=P(t)s(t)\left[s(t)^TP(t)s(t)+1\right]^{-1}\\
&P(t+1)=\left[I-q(t)s(t)^T\right]P(t)\\
&\theta_{t+1}^{(i)}=\theta_t^{(i)}+q(t)\left[z(t)-s(t)^T\theta_t^{(i)}\right]
\end{aligned} \tag{7}$$

3.3 Policy Improvement with Policy Approximation

After $\theta^{(i)}$ is achieved, policy improvement continues to extract an improved policy $\hat{h}^{(i+1)}$. However, because of the linear parametrization for Q function and polynomials BFs, it is difficult to solve policy improvement (5) directly and have an explicit solution of $\omega^{(i+1)}$ associated with $\hat{h}^{(i+1)}$. In this way, a gradient-based method is more suitable for policy improvement, which is denoted by $\omega_j=\omega_j+\alpha\frac{\partial Q}{\partial u_j}\frac{\partial u_j}{\partial \omega_j}$, where α is the learning rate.

To guarantee the accuracy of $\hat{h}^{(i+1)}$ or $\omega^{(i+1)}$ on the whole state space Ω, a training set which is evenly distributed over the state space is defined beforehand, $\{X_s\}$, $s=1,...,N_s$. For any BF ϕ_i, its partial differential to action u_j has the following form

$$\frac{\partial \phi_i}{\partial u_j}(x,u)=\begin{cases}\beta_j^i \dfrac{\phi_i(x,u)}{u_j}, & \text{if } u_j \neq 0\\ 0, & \text{if } u_j = 0\end{cases} \tag{8}$$

So partial differential of $\hat{Q}^{(i)}$ can be formulated by $\frac{\partial \hat{Q}^{(i)}}{\partial u_j}(x,u)=\phi_{u_j}^T(x,u)\theta^{(i)}$ where $\phi_{u_j}=\left[\frac{\partial \phi_1}{\partial u_j},...,\frac{\partial \phi_N}{\partial u_j}\right]^T$. In this way, the gradient-based updating formula for $\omega_j^{(i+1)}$ on training set $\{X_s\}$ is obtained

$$\omega_{j,s}^{(i+1)}=\omega_{j,s-1}^{(i+1)}+\alpha\phi_{u_j}^T(X_s,\varphi^T(X_s)\omega_{j,s-1}^{(i+1)})\theta^{(i)}\varphi(X_s) \tag{9}$$

where $j=1,...,m$. It is noted that in order to generate an accurate parameter $\omega^{(i+1)}$, updating formula (9) on training set $\{X_s\}$ can be implemented for sufficient times.

3.4 Exploration

Exploration is necessary for online algorithm to find optimal polices. Here, we introduce ε-greedy exploration and reset scheme in [11]. At each step t, it has $1-\varepsilon_t$ probability to apply the current policy directly and ε_t probability to add uniform random exploration noise n_t to the action. Besides, at the beginning of the algorithm, the exploration probability is relatively large to encourage exploration. As the algorithm runs, the proportion of the exploitation increases. A decay exploration is denoted by $\varepsilon_t=\varepsilon_0\varepsilon_d^t$, where ε_0 is the initial value and ε_d is the decay factor with $0<\varepsilon_d<1$.

However, as the policy is admissible and the exploration noise is small, the system can still be stabilized after enough steps. At that time, the added exploration noise is not sufficient to drive the state away from the equilibrium and a new trial starting from non-equilibrium points is more benefit for the exploration. Namely, after every T_{trial} steps, the state is reset away from equilibrium.

Algorithm 1 presents our online RLSPI algorithm. It should be noted that during the implementation, no information of system dynamics is needed and the algorithm only relies on online data.

Algorithm 1. Online RLSPI algorithm

initialize $\theta^{(0)} \leftarrow 0$, $\omega^{(0)} \leftarrow h_0$, $P(0) = aI_{N\times N}$, $i = 0$ and x_0
for every step $t = 0,1,2,...$ do
$$u_t = \begin{cases} \hat{h}^{(i)}(x_t) & \text{at } 1-\varepsilon_t \text{ probability} \\ \hat{h}^{(i)}(x_t) + n_t & \text{at } \varepsilon_t \text{ probability} \end{cases}$$
apply u_t and measure next state x_{t+1} and reward r_t
policy evaluation $\theta^{(i)}$ by (7)
if $t = (i+1)K_{update}$ then
policy improvement $\omega^{(i+1)}$ using (9) on $\{X_s\}$ and $i = i+1$
end if
end for

4 Simulation Example

The nonlinear discrete-time system to test our algorithm is a mass-spring system with two states and one action. The dynamics of this system is presented in [14]. We define the reward function by a negative definite quadratic function with respect to state and action, $r(x_k, u_k) = -x_k^T Q x_k - u_k^T R u_k$, where $Q = 0.5I_{2\times 2}, R = 1$.

As the system is nonlinear, the associated Q function or policy is complicated and a plenty of BFs are required to achieve high precision. In this way, up to 6-order $\phi(x,u)$ and up to 5-order $\varphi(x)$ are used

$$\varphi(x,u) = \begin{bmatrix} x_1^2, x_2^2, u^2, x_1x_2, x_1u, x_2u, x_1^4, x_2^4, u^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, u[x_1^3, x_1^2x_2, x_1x_2^2, x_2^3], \\ u^2[x_1^2, x_1x_2, x_2^2], u^3[x_1, x_2], x_1^6, x_2^6, u^6, x_1^5x_2, x_1^4x_2^2, x_1^3x_2^3, x_1^2x_2^4, x_1x_2^5, \\ u[x_1^5, x_1^4x_2, x_1^3x_2^2, x_1^2x_2^3, x_1x_2^4, x_2^5], u^2[x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4], \\ u^3[x_1^3, x_1^2x_2, x_1x_2^2, x_2^3], u^4[x_1^2, x_1x_2, x_2^2], u^5[x_1, x_2] \end{bmatrix}^T$$

$$\varphi(x) = \left[x_1, x_2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^5, x_1^4x_2, x_1^3x_2^2, x_1^2x_2^3, x_1x_2^4, x_2^5 \right]^T$$

For this system, the initial state of each trial is set to $[-0.2, 0.2]^T$. And the length of each trial T_{trial} is 1000 steps. Besides, the policy evaluation length K_{update} is set to 200 and the training set $\{X_s\}$ selects $\{-0.1, -0.08, ..., 0.08, 0.1\}^2$. The uniform random exploration noise is limited between $[-0.1, 0.1]$. The initial exploration value ε_0 is 1 and the decay factor ε_d is 0.999977. As the system is self-stable, policy equal to 0 is adopted as the initial admissible policy.

The results of the RLSPI algorithm is presented in Fig. 1, where (a) reveals 10 representative trials during implementation, respectively at 0^{th}, 10000^{th}, 20000^{th}, …,

90000^{th} step. It is obvious that our algorithm can learn to improve the policy to a good one which stabilizes the system very well. The performance of the finial learned policy using RLSPI is presented in (b) starting from $[-0.2, 0.2]^T$. And (c) reveals scores of policies at the end of different trials in RLSPI with respect to step index. The scores of policies increase as RLSPI running.

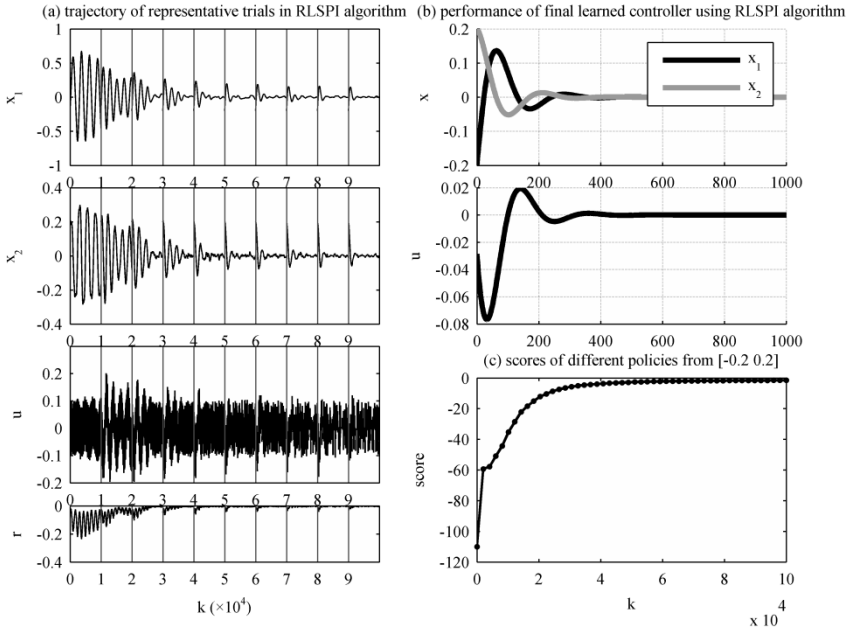


Fig. 1. Implementation of RLSPI algorithm for nonlinear discrete-time system. (a) Trajectory of representative trials in RLSPI algorithm. (b) Performance of finial learned policy using RLSPI algorithm. (c) Scores of different policies from $[-0.2, 0.2]^T$.

5 Conclusion

PI method for nonlinear discrete-time non-affine systems with continuous-state and continuous-action space is considered in this paper. Q function is introduced to approach undiscounted value function in PI. Relying on Q function, PI method does not need the information of system dynamics. Using linear parametrization, an online model-free RLSPI algorithm is proposed with RLS method and continuous policy approximation. The algorithm reveals efficiency on nonlinear discrete-time systems. Without the information of system dynamics and only relying on online data, RLSPI can learn optimal or near-optimal policies with finite steps.

References

1. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
2. Wang, F., Zhang, H., Liu, D.: Adaptive Dynamic Programming: An Introduction. IEEE Comput. Intell. Mag. 4(2), 39–47 (2009)
3. Lewis, F.L., Vrabie, D.: Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control. IEEE Circuits Syst. Mag. 9(3), 32–50 (2009)
4. Howard, R.: Dynamic Programming and Markov Processes. MIT Press, Cambridge (1960)
5. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific, Belmont (1996)
6. Tsitsiklis, J.N., Van Roy, B.: Feature-Based Methods for Large Scale Dynamic Programming. Machine Learning 22, 59–94 (1996)
7. Tsitsiklis, J.N., Van Roy, B.: An Analysis of Temporal Difference Learning with Function Approximation. IEEE Trans. Automat. Contr. 42(5), 674–690 (1997)
8. Zhao, D.B., Bai, X.R., Wang, F.Y., Xu, J., Yu, W.S.: DHP Method for Ramp Metering of Freeway Traffic. IEEE Transactions on Intelligent Transportation Systems 12(4), 990–999 (2011)
9. Zhao, D.B., Hu, Z.H., Xia, Z.P., Alippi, C., Wang, D.: A Human-Like Full Range Adaptive Cruise Control Based on Supervised Adaptive Dynamic Programming. Neurocomputing (in press), http://dx.doi.org/10.1016/j.neucom.2012.09.034
10. Si, J., Wang, Y.T.: On-Line Learning Control by Association and Reinforcement. IEEE Trans. Neural Netw. 12(2), 264–276 (2001)
11. Busoniu, L., Ernst, D., De Schutter, B., Babuska, R.: Online Least-Squares Policy Iteration for Reinforcement Learning Control. In: Proc. 2010 American Control Conf. (ACC 2010), pp. 486–491 (2010)
12. Lagoudakis, M.G., Parr, R.: Least-Squares Policy Iteration. Journal of Machine Learning Research 4, 1107–1149 (2003)
13. Abu-Khalaf, M., Lewis, F.L.: Nearly Optimal Control Laws for Nonlinear Systems with Saturating Actuators Using a Neural Network HJB Approach. Automatica 41(5), 779–791 (2005)
14. Zhang, H., Luo, Y., Liu, D.: Neural-Network-Based Near-Optimal Control for a Class of Discrete-Time Affine Nonlinear Systems with Control Constraints. IEEE Trans. Neural Netw. 20(9), 1490–1503 (2009)

An Entropy Based Method for Causal Discovery in Linear Acyclic Model*

Yulai Zhang and Guiming Luo

Tsinghua National Laboratory for Information Science and Technology
School of Software, Tsinghua University, Beijing 100084 China
{zhangyl08,gluo}@mail.tsinghua.edu.cn

Abstract. A new algorithm for causal discovery in linear acyclic graphic model is proposed in this paper. The algorithm measures the entropy of observed data sequences by estimating the parameters of its approximate distribution to a generalized Gaussian family. Causal ordering can be discovered by an entropy base method. Compared with previous method, the sample complexity of the proposed algorithm is much lower, which means the causal relationship can be correctly discovered by a smaller number of samples. An explicit requirement of data sequences for correct causal inference in linear acyclic graphic model is discussed. Experiment results for both artificial data and real-world data are presented.

Keywords: Causal inference, Linear Acyclic Model, Entropy.

1 Introduction

Causal discovery from observed data is an important problem in various domains. A lot of works have been done in the field of machine learning. A linear non-Gaussian acyclic model for casual discovery is proposed in [10] in which an independent component analysis technic is used to discover the causal relations of data sets. Following this line of research, a model for causal inference from data sequences with time structures is proposed in [4]. [5] extended the acyclic graph model method to a cyclic one. And nonlinear causal models were discussed in [3] and [1].

In above works, the information for correct causal discovering comes from either non-Gaussian noises or nonlinear functions [1], or both [3]. Once the causal direction is found, the rest of work is data modeling, which is a traditional topic and can be done by a variety of methods. As pointed out in [9], the effect of nonlinear function or additive noises is making the result sequence closer to a standard normal distribution. In generalized Gaussian distribution family this distance can be measure by the entropy of the probability density function with unit second order central moment. Using this feature implicitly, the LiNGAM method [10] requires non-gaussian data sequences. When noise signal ratio is low, the sample complex of this algorithm increases rapidly. Unfortunately, sample data collection can be a laborious job in many applications.

In this paper, we first discussed the conditions for correct causal inference, which is influenced by noises pdf, noise signal ratio, model parameters and so on. Based on this

* This work is supported by the Funds NSFC61171121 and the Science Foundation of Chinese Ministry of Education-China Mobile 2012.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 250–256, 2013.

discussion, a new algorithm for causal discovery on the model of linear acyclic graph with additive noises is proposed. The entropy of an equivalent generalized Gaussian distribution is used to measure the gaussianity of sequences. The propose entropy based algorithm has a much lower sample complexity than the previous method, especially in the cases with small additive noises. When both the signal sequences and the noise sequences are pure Gaussian, the entropy value will be equal for both directions thus wrong results can be avoided.

2 Preliminaries

2.1 Linear Acyclic Model

Linear acyclic model is an important way to present causal relationships among observed variables. $x_i, i \in \{1, ..., m\}$are m data sequences with length n. X is an $m \times n$ matrix. Each row of X is a sequence of one observed variable. The causal ordering of these variable can be denoted by an order function $k(x_i)$. $e_i, i \in \{1, ..., m\}$ are the correspond noise data. The linear acyclic model with additive noise can be expressed as: $x_i = \sum_{k(x_j)<k(x_i)} b_{ij} x_j + e_i$.

Definition 1. *For sequence e_i,if $b_{ij} = 0$ for all j, e_i is called a signal sequence and the otherwise e_i is a noise sequence.*

2.2 Generalized Gaussian Distribution(GGD)

The generalized gaussian distribution's PDF is: $f(x) = \frac{s}{2\sigma\Gamma(1/s)} exp\{-|\frac{x-\mu}{\sigma}|^s\}$. For simplicity we discuss the GGD with $\mu = 0$. The details of GGD's parameters and properties can be find in [7]. Many widely used distributions can be represented or approximated by generalize gaussian distribution.

The generation of GGD data is discussed in [8]. A maximum likelihood estimator for parameters of GGD is proposed in [2]. The shannon entropy of the GGD is defined as: $Entropy(f(x|s, \sigma)) = \frac{1}{s} - log\{\frac{s}{2\sigma\Gamma(1/s)}\}$ The entropy of a random variable with a particular distribution is influenced by its shape of a pdf as well as the value of its variance. $var(f(x|s, \sigma)) = \frac{\sigma^2\Gamma(3/s)}{\Gamma(1/s)}$ If the variance or the second order moment of the pdf is set to be 1, that is $\sigma^2 = \Gamma(1/s)/\Gamma(3/s)$, we can obtain the following entropy function for GGD with unit variance.

Definition 2. $UnitEntropy(s) = \frac{1}{s} - log\{\frac{s\Gamma(3/s)^{\frac{1}{2}}}{2\Gamma(1/s)^{\frac{3}{2}}}\}$ *This is entropy function for GGD with unit variance.*

3 Main Results

In this section, we first discuss the conditions for two variables in the linear model using entropy of GGD. Then we extend the results to a graphic model with more than 2 variables.

Lemma 3. *The entropy function for GGD with unit variance* $UnitEntropy(s)$ *reaches the highest value when* $s = 2$ *(a standard normal distribution).*

Though this is an obvious result, the derivative may not be straightforward. We calculated the value of $UnitEntropy(s)$. It increases rapidly for s from near 0 to 2, and decreases very slowly from 2 to infinity.

Lemma 4. x_1 *and* x_2 *are random variables with unit variance.* x_1 *is non-gaussian.* $y = ax_1 + (1-a^2)^{\frac{1}{2}}x_2$, $|a| < 1$ *apparently* $var(y) = 1$, *define* $\gamma = \frac{a^2}{1-a^2}$ *as the signal noise ratio. Suppose* x_1 *is the cause and* x_2 *is the noise and* y *the effect. The effect sequence* y *is closer to standard gaussian than the cause sequence* x_1*, if:*

$$h(\gamma) = (1+\gamma)^{n/2} - \gamma^{n/2} > \kappa_n(x_2)/\kappa_n(x_1) \tag{1}$$

where n *is even integers larger than 2 and* $\kappa_n(x)$ *is the* nth *cumulant for* x*.*

Proof. For a standard gaussian distribution (GGD with $s = 2$) the $\kappa_n = 0\ \forall n \geq 4$; And for GGD $s \neq 2$,$\kappa_n = 0$ when n is an odd number and κ_n is positive when n is an even number. So if $\kappa_n(y) < \kappa_n(x_1), \forall n \geq 4$, we can say that y is closer to gaussian than x_1

$$\kappa_n(y) = \kappa_n(ax_1 + \sqrt{1-a^2}x_2) = a^n\kappa_n(x_1) + (1-a^2)^{n/2}\kappa_n(x_2)$$

when $\kappa_n(y) < \kappa_n(x_1)$, we have $(1-a^n)\kappa_n(x_1) > (1-a^2)^{n/2}\kappa_n(x_2)$ where $\kappa_n(x_1)$ and $\kappa_n(x_2)$ are positive for even number of $n \geq 4$. And $(1-a^n)/(1-a^2)^{n/2}) > \kappa_n(x_2)/\kappa_n(x_1)$ notice that $a^2 = \frac{\gamma}{1+\gamma}$, then the equation (1) is obtained.

Theorem 5. x_1 *and* x_2 *are random variables with arbitrary variance, and* x_1 *is non-gaussian.* $y = bx_1 + x_2$*, where* x_1 *and* y *are input and output of a linear causal relationship respectively .* x_2 *is the disturb noise. The causal direction between* x_1 *and* y *can be correctly discovered by comparing the unit entropy function when the following inequality holds:*

$$h(\gamma) > \frac{\kappa_n(x_2/\sqrt{v_2})}{\kappa_n(x_1/\sqrt{v_1})} \tag{2}$$

where $h(\gamma) = (1+\gamma)^{n/2} - \gamma^{n/2}$ *and* $\gamma = b^2v_1/v_2$ v_1 *and* v_2 *are variance of* x_1 *and* x_2*,* $n \geq 4$*.*

Proof. Theorem 5 can be directly obtained using Lemma 3 and Lemma 4.

If the condition in theorem 5 is met, the unit entropy function value of the cause data sequence is smaller than the value of the effect data sequence. Suppose s_{x_1} and s_y are the parameter of the approximate GGD of x_1 and y, $UnitEntropy(s_{x_1}) < UnitEntropy(s_y)$ when x_1 is the cause and y is the effect.

Corollary 6. x_1 *and* x_2 *are random variables with arbitrary variance.* $y = bx_1 + x_2$*, where* x_1 *and* y *are input and output of a linear causal relationship respectively .* x_2 *is the disturb noise. If* x_2 *is more gaussian than* x_1*, the causal direction can be correctly obtained.*

Proof. Under conditions in Corollary 6, the value of right side of (2) is in $(0, 1)$, so (2) always holds.

Next, we extend the result into model with more than 2 sequences. If the causal model is a tree like graphic model, the above theorems can be directly applied. Otherwise if the effect variable has more than one causes, rewrite graph model equation as: $x_i = b_{ij}x_j + \bar{e}_i$ where $\bar{e}_i = \sum_{k(x_l)<k(x_i), l\neq j} b_{il}x_l + e_i$ In a linear acyclic model, The causal order between x_i and x_j can be correctly discovered by comparing the unit entropy function when the following inequality holds: $h(\gamma) > \frac{\kappa_n(\bar{e}_i/\sqrt{v_{\bar{e}_i}})}{\kappa_n(x_j/\sqrt{v_{x_j}})}$ and $\gamma = b_{ij}^2 v_{x_j}/v_{\bar{e}_i}$ v_{x_j} and $v_{\bar{e}_i}$ are variance of x_j and $\bar{e}_i$, and $\forall n \geq 4$.

4 Algorithm for Causal Discovery in Acyclic Graphic Model with Additive GGD

Based on the results of previous section, we propose an algorithm for causal discovery in acyclic graph with additive generalized gaussian noises. $x_1, ..., x_m$ are observed data vectors with length n. X is the $m \times n$ data matrix $X = (x_1, ..., x_m)^T$. $e_1, ..., e_m$ are source data sequence in this model, $S = (e_1, ..., e_m)^T$. So the vector version of linear acyclic model can be write as: $X = BX + S$ and $X = AS$ where $A = W^{-1} = (I - B)^{-1}$, I is a unit matrix. B is a lower triangle matrix defined directly by the parameters b_{ij} of the graphic model. And $W = I - B$ is a lower triangle matrix with zeros on the main diagonal.

Algorithm of Entropy Based Method for causal discovery:

STEP1:Estimate parameters of an approximate GGD. Let $u_i = (x_i - \mu_{x_i})/\sqrt{var(x_i)}$. Estimate the parameter of the approximate GGD probabilistic density function of $u_i, i = 1...m$. The probability density function of the GGD can be expressed as $f_i(x|\sigma_i, s_i)$. σ_i and s_i are obtained.

STEP2:Calculate the unit entropy of x_i. $en_i = UnitEntropy(s_i)$. Sort the vector of unit entropy en in ascending order. The ordering function $k(x_i)$ can be obtained by this order. Let $k(x_i)$ equals the index of unit entropy en_i in the sorted vector.

STEP3:Rearrange the rows of matrix X. $\bar{X} = PX$ where P is the row permutation matrix of X, with $P_{k(x_i)i} = 1$ and other elements of P is zero. The matrix $\bar{X}$ contains the observed variables in the entropy order.

STEP4:Decomposition $\bar{X}$. Apply an independent component analysis (ICA) algorithm to matrix $\bar{X}$. Matrix W as well as matrix A described above can be obtained.

STEP5:Find the right order of e_i by sloving: $\bar{W} = \min_{\bar{w}\in Perm(W)} \sum_i \frac{1}{|W_{ii}|^{s_i}} Perm(W)$ is a set comprises all the row permutation matrix of W.

STEP6:Compute the matrix $\hat{W}$ where the elements of $\hat{W}$ is denoted as $\hat{w}_{ij}$ and let

$$\hat{w}_{ij} = \begin{cases} 0 & \text{for } i < j \\ \bar{w}_{ij}/\bar{w}_{ii} & \text{for } i \geq j \end{cases}$$

The diagonal elements of $\hat{W}$ is 1.

STEP7: The estimation of matrix B is given by $\hat{B} = I - \hat{W}$

In the above algorithm step 4, 6, 7 are similar to LiNGAM while step 1-3 and 5 are different.

5 Experiments

Three experiments with simulated data are presented to illustrate the method's performance under different s, γ, and m. The performance of the proposed Entropy Based Method and LiNGAM in [10] are both affected by these parameters. All the experiments are done by Matlab. We generate sequence e_i in generalized gaussian distribution by the method in [8].

The results of first experiment are shown in figure 1. In this experiment we generate generalized Gaussian density noises with $s = 0.5, 1.0, 1.5, 2.0, 2.5, 3.0$. The number of observed variable is $m = 2$, both e_i in this experiment have same s. For the other parameters, we set $\gamma = 1.25$ with $b_{21} = 0.5$. The X axis is the length of an observed variable n with ranges from 200 to 2000. The Y axis is the percentage of the experiments with causal order function $k(x_i)$ correctly calculated. For each point in the figure, 100 experiments are done with random generated observed data sets.

The Entropy Based Method converges and gives a better performance, except when $s = 2$, which means that all data sequences are pure gaussian. But the result given by Entropy Based Method under pure gaussian is more reasonable than LiNGAM. The chance is 50-50 for candidate causal orders. By contrast, the result of LiNGAM can be misleading. As can be seen from the figure, sample complexity increases for both method when s is close to 2.

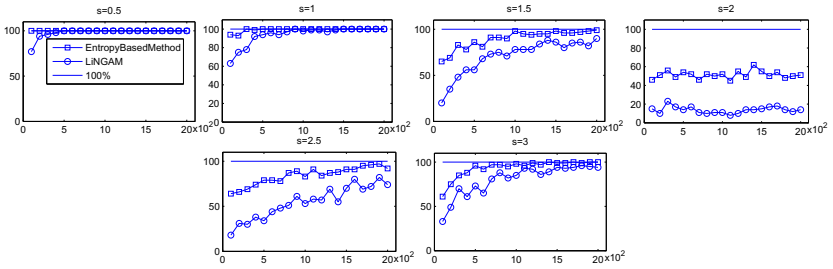


Fig. 1. Performance of Causal Discovery Methods on different s. X-axis: length of the observed variable n, from $n = 200$ to 2000; Y-axis: The number of successes in 100 experiments.

In the experiment represented in figure 2, the performance of Entropy Based Method and LiNGAM under different value of γ are compared. γ represents the ratio of signal and noise in the linear equation. We let $\gamma = 2.5, 1.25, 0.5, 0.25$ and $s = 1$. The rest parameters are the same as first experiment. The performance of Entropy Based Method is better than LiNGAM under different value of γ.

In the third experiment, we compare the performance of methods' when $m = 3$ and 4, where m is the number of observed variables.

For $m = 3$, we set $b_{21} = 0.4$, $b_{31} = 0.1$, and $b_{32} = 0.6$. The rest elements in matrix B are set to be 0, so B is a low triangle matrix. The variance of $e_1, e2$ and e_3 are all set to be 1. And parameter s of $e_1, e2$ and e_3 are $0.5, 1.5, 2.0$ respectively. For $m = 4$,

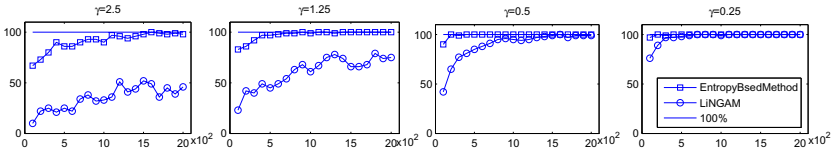


Fig. 2. Performance of Causal Discovery Methods on different γ

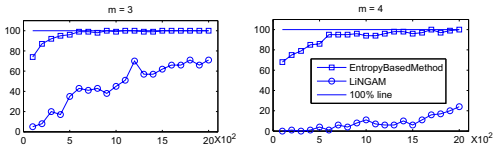


Fig. 3. Performance of Causal Discovery Methods on different number of observed sequences m

we set $b_{21} = 0.5$, $b_{31} = 0.1$, $b_{32} = 0.4$, $b_{41} = 0.0$, $b_{42} = 0.1$, and $b_{43} = 0.2$. The variance of $e_i, i = 1...4$ are set to be $1.0, 1.0, 0.8, 0.8$ respectively.

The time complexity of the entropy based method is mainly affected by the sample size $m \times n$, where n is the length of an observed data sequence and n is number of data sequence. The running time of the Entropy Based Method is generally equal to the previous method.

In experiment with real-world data, abalone data set in [6] is analyzed. This data set can also be found in UCL repository of machine learning. Though nonlinear causal modeling for this data set is also discussed in [3], linear modeling is more practical in many applications.

Abalone is a kind of shellfish. Data sequence X denotes the number of rings on a shell, and Y is the diameter of the shell. Number of rings equals to the age of the shellfish plus 1.5. So the causal order should be X causes Y, or $k(X) < k(Y)$, if presented in an order function.

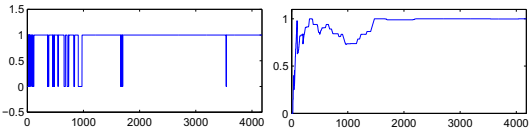


Fig. 4. Performance of Entropy Based Method on Abalone data set. Left: Result of the causal discovery using first x samples, 1 indicates success; Right: Moving average of the top half of the figure with a window length of 500.

In the left half of figure 4, the x-axis mean the length of vector, n. The value of y-axis denotes whether the correct causal order is obtained. Value 1 means a correct causal inference and 0 mean an incorrect one. As shown in figure 4, the Entropy Based Method makes a stable choice after 1500 number of samples.

By contrast, the LiNGAM algorithm gives the wrong causal order in every sample size from 20 to above 4000. This Entropy based method works well on abalone data for several reasons.The data distribution of abalone data is somewhat close to GGD and the noise in this data set can be interpreted as additive. Though we believe the causal relation is actually nonlinear, it can also be interpreted by a linear model.

6 Conclusions

In this paper, a new method for causal discovery in linear acyclic model is proposed. This algorithm finds the causal order using an entropy based method. The conditions to get correct causal orders in the linear acyclic model are discussed. In a linear acyclic model the result of causal discovery can be affected by the distribution of signals and noises sequences, the signal noise ratio, the number of sample variables and its length. Experiments results on both artificial data and the real-world data set abalone are presented. The Entropy Based Method gives better experimental result than the state-of-art method. The sample complexity of this new method is significantly lower. Since data collection can be a hard work in many field, a method with lower sample complexity is valuable.

References

1. Daniusis, P., Janzing, D., Mooij, J., Zscheischler, J., Steudel, B., Zhang, K., Schoelkopf, B.: Inferring deterministic causal relations. In: Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2010 (2010)
2. Do, M.N., Vetterli, M.: Wavelet-based texture retrieval using generalized gaussian density and kullback cleibler distance. IEEE Transaction in Image Processing 11(2), 146–158 (2006)
3. Hoyer, P.O., Janzing, D., Mooij, J., Peters, J., Scholkopf, B.: Nonlinear causal discovery with additive noise models. In: Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems (NIPS 2008), Vancouver (December 2008)
4. Hyvarinen, A., Shimizu, S., Hoyer, P.O.: Causal modelling combining instantaneous and lagged effects: an identifiable model based on non-gaussianity. In: Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland (2008)
5. Lacerda, G., Spirtes, P.L., Ramsey, J., Hoyer, P.O.: Discovering cyclic causal models by independent components analysis. In: Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2008 (2008)
6. Mooij, J., Janzing, D.: Distinguishing between cause and effect. Journal of Machine Learning Research Workshop & Conference Proceedings 6, 147–156 (2010)
7. Nadarajah, S.: A generalized normal distribution. Journal of Applied Statistics 32(7), 685–698 (2005)
8. Nadarajah, S., Kotz, S.: On the generation of gaussian noise. IEEE Transaction in Signal Processing 55(3), 1172 (2007)
9. Pearl, J.: Causality: Models, Reasoning, and inference. Cambridge University Press (2008)
10. Shimizu, S., Hoyer, P.O., Hyvarinen, A., Kerminen, A.: A linear non-gaussian acyclic model for causal discovery. Journal of Machine Learning Reseach 7, 2003–2030 (2006)

Similarity Measures for Link Prediction Using Power Law Degree Distribution

Srinivas Virinchi* and Pabitra Mitra

Dept of Computer Science and Engineering, Indian Institute of Technology Kharagpur-721302, India
{virinchimnm,pabitra}@cse.iitkgp.ernet.in

Abstract. Link Prediction deals with predicting important non-existent edges in a social network that are likely to occur in the near future. Typically a link between two nodes is predicted if the two nodes have high similarity. Well-known local similarity functions make use of information from all the local neighbors. These functions use the same form irrespective of the degree of the common neighbors. Based on the power law degree distributions that social networks generally follow, we propose non-linear schemes based on monotonically non-increasing functions that give more emphasis to low-degree common nodes than high-degree common nodes. We conducted experiments on several benchmark datasets and observed that the proposed schemes outperform the popular similarity function based methods in terms of accuracy.

Keywords: Degree of neighbors, Common Neighbors, Adamic Adar, Resource Allocation Index, Markov Inequality.

1 Introduction

Link prediction problem employs similarity that could be based on either local or global neighborhood. Here, we concentrate on some of the state-of-the art local neighborhood measures. One of the local measures which is popularly employed is the Resouore Allocation Index. The similarity function employed by this index makes the contribution of each of the common neighbors inversely proportional to its degree.

Social networks follow the power law degree distribution. According to this law, the probability of encountering a high degree node is very small. We use an appropriate threshold to split the set of nodes based on their degree into low and high degree node sets. We have given different weights to common nodes in different clusters while computing the similarity between a pair of nodes. Specifically, we have compared the performance of the modified algorithms with common neighbors, Adamic Adar and resource allocation index. We emphasize the role of low degree common nodes in terms of their contribution to the similarity and either deemphasize or ignore the contributions of high degree common

* Corresponding author: virinchimnm@gmail.com

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 257–264, 2013.

nodes. We justify the proposed scheme formally. The modified algorithms have resulted in an improved performance in terms of classification accuracy on several benchmark datasets. Our specific contributions in this paper are:

1. We establish formally that the number of high degree common neighbors is insignificant compared to the low degree common neighbors.
2. To deemphasize the role of high degree neighbors and increase the contribution of low degree neighbors in computing similarity.

2 Background and Related Work

We can view any social network as a graph G. Each link can be viewed as an edge in the graph. We can represent the graph as G = (V, E) where V is the set of vertices and E is the set of edges in the graph. Now, let us consider that at some future instance t' the graph after addition of some edges has become $G' = (V', E')$ where V' is the set of vertices of graph G' and E' is the set of edges of G'. Link Prediction problem deals with the prediction of edges from the set of edges $E' - E$ accurately.

According to the power law, the probability of finding a k degree node in the social network which is denoted by p_k is directly proportional to $k^{-\alpha}$ where α is some positive constant usually between 2 and 3. Hence, the probability of finding a high degree node in the graph is very less as the corresponding value of k is very high.In general, given G, it is not clear as to which are high degree nodes and which are low degree nodes. For differentiating between high degree and low degree nodes we make use of the Markov Inequality. It can be defined as: if X is a non-negative random variable and there exists some positive constant $b > 0$ then,

$$P(X \geq b) \leq \frac{E[X]}{b}. \tag{1}$$

We make use of the above inequality to find a threshold value (T) which divides the set of nodes based on degrees into low degree nodes and high degree nodes. In this case, as degree is always non-negative we can make degree as the non-negative random variable and T will be positive, we can represent the above inequality as:

$$P(degree \geq T) \leq \frac{E[degree]}{T} \Rightarrow T \leq \frac{E[degree]}{P(degree \geq T)}. \tag{2}$$

Thus, from inequality (2) we can calculate the threshold that we require based on the number of high degree nodes that we need. The similarity functions could be either local or global (takes the whole graph into account along with some distance based measures). [1] and [2] present the survey of various similarity measures and higher level approaches used in the area of link prediction. Further, they show that Resource Allocation Index outperforms the other local similarity measures. [3] refines the common neighbors approach by giving the less connected neighbors higher weight and is popularly known as the Adamic-Adar index (AA)

known after the authors. The authors in [4] designed the Resource Allocation (RA) Index which is motivated by the resource allocation dynamics where node x can transmit a resource through a common neighbor to another node y. In [5], the authors make use of the latent information from communities and showed that embedding community information into the state-of-the-art similarity functions can improve the classification accuracy.

3 Standard Similarity Measures and Our Approach

We formally explain different similarity functions which are used for experimental comparison against our approach. Here, $N(x)$ is the set of nodes adjacent to node x.

Common Neighbors (CN): Score between nodes x and y is calculated as the number of common neighbors between x and y.

$$CN(x, y) =\mid N(x) \cap N(y) \mid . \tag{3}$$

Adamic Adar (AA): Score between nodes x and y is calculated as the sum of inverse of the log of degree of each of the common neighbors z between x and y.

$$AA(x, y) = \sum_{z \in N(x) \cap N(y)} \frac{1}{log(degree(z))}. \tag{4}$$

Resource Allocation Index (RA): Score between nodes x and y is calculated as the sum of inverse of the degree of each of the common neighbors z between x and y.

$$RA(x, y) = \sum_{z \in N(x) \cap N(y)} \frac{1}{degree(z)}. \tag{5}$$

We use (2) discussed in section 2 to bound the threshold. For conducting the experiment, we use $P(degree \geq T)$ between 0.01 and 0.1 and on experimenting we find the best threshold for each dataset. The thresholds are listed in table 2. Once we have the threshold value, we divide the node set V based on the threshold into low degree and high degree node sets. We justify our deemphasis of the high degree common neighbors using the theorem below. Let,

- n = Number of nodes in the graph, n_L = Number of Low degree nodes, n_H = Number of High Degree nodes, p_k = Probability existence of a k degree node in graph
- L_{avg} = Average degree of a node in the low degree region , H_{avg} = Average degree of a node in the high degree node
- K_L = Expected number of low degree common neighbors for any pair of nodes, K_H = Expected number of low degree common neighbors for any pair of nodes
- T = Threshold on degree, max = Maximum degree, $L = \{x|degree(x) < T\}$, $H = \{x|degree(x) \geq T\}$

Theorem 1. *For any pair of nodes x and y, the expected number of common neighbors of high degree is very small when compared to expected number of common neighbors of low degree.*

Proof: Consider the possibility that both $x, y \in L$. The probability that a common neighbour $z \in L$ is given by

$$P(z \in L) = \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}}. \tag{6}$$

where $\frac{n_L}{n}$ accounts for the selection of a node from L and $p_{L_{avg}}$ accounts for the average probability of existence of a low degree node. Note that the first four terms correspond to the probability of existence of an edge between x and z and the last four terms correspond to the probability of existence of an edge between y and z. The above equation can be simplified to the following form :

$$P[z \in L] = \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}} * (\frac{n_L}{n} * p_{L_{avg}})^2. \tag{7}$$

In a similar way the probability that a common neighbor $z \in H$ is given by

$$P[z \in H] = \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}} * (\frac{n_H}{n} * p_{H_{avg}})^2. \tag{8}$$

So,

$$K_L = n * P[z \in L] = n * \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}} * (\frac{n_L}{n} * p_{L_{avg}})^2. \tag{9}$$

and

$$K_H = n * P[z \in H] = n * \frac{n_L}{n} * p_{L_{avg}} * \frac{n_L}{n} * p_{L_{avg}} * (\frac{n_H}{n} * p_{H_{avg}})^2. \tag{10}$$

Thus, the ratio of K_H to K_L, from (4) and (5) is given by

$$\frac{K_H}{K_L} = (\frac{n_H}{n_L})^2 * (\frac{p_{H_{avg}}}{p_{L_{avg}}})^2. \tag{11}$$

Note that there are 3 other possbilities for assigning x and y to L and H; these are: 1. $x \in L$ and $y \in H$, 2. $x \in H$ and $y \in L$ and 3. $x \in H$ and $y \in H$ Note that in all these 3 cases also, the value of $\frac{K_H}{K_L}$ is the same as the one given in (6). Using power law

$$p_{L_{avg}} = C * (L_{avg})^{-\alpha}. \tag{12}$$

From (6) and (7), we have

$$\frac{K_H}{K_L} = (\frac{n_H}{n_L})^2 * (\frac{L_{avg}}{H_{avg}})^{2\alpha}. \tag{13}$$

Consider $\frac{n_H}{n_L}$ which can be simplified as, (degree 1 nodes are ignored)

$$\frac{n_H}{n_L} = \frac{n * \sum_{i=T}^{max} p_i}{n * \sum_{i=2}^{T-1} p_i}. \tag{14}$$

Note that

$$2^{-\alpha} \leq \sum_{i=2}^{T} i^{-\alpha} \leq (T-1) * 2^{-\alpha}. \tag{15}$$

Using power law we can substitute $i^{-\alpha}$ for p_i and cancelling out n, we get

$$\frac{n_H}{n_L} = \frac{\sum_{i=T}^{max} i^{-\alpha}}{\sum_{i=2}^{T-1} i^{-\alpha}} \leq \frac{(max-T) * T^{-\alpha}}{2^{-\alpha}} = (max-T) * (\frac{2}{T})^{\alpha}. \tag{16}$$

Also by noting that $L_{avg} < T$ and $H_{avg} > T$ we can bound $\frac{L_{avg}}{H_{avg}}$ as follows,

$$\frac{L_{avg}}{H_{avg}} = \frac{T-\delta}{T+\delta} \; for \; some \; 1 < \delta < max \tag{17}$$

$$\frac{L_{avg}}{H_{avg}} = (1 - \frac{2\delta}{T+\delta}) < e^{-}\frac{2\delta}{T+\delta} \tag{18}$$

So from (13), (16) and (18), we get

$$\frac{K_H}{K_L} \leq (max-T)^2 * (\frac{2}{T})^2 * (e^{-}\frac{2\delta}{T+\delta})^{2\alpha} \tag{19}$$

which is a very small quantity and it tends to zero as T tends to a large value which happens when the graph is large. It is intuitively clear that $L_{avg} < H_{avg}$. Further, because of the power law and selection of an appropriate threshold value we can make $\frac{n_H}{n_L}$ as small as possible. For example, by selecting the value of T to be less than or equal to $\frac{max}{2}$ where $max = max_x(degree(x))$ we get $\frac{n_H}{n_l}$ to range between 0.003 to 0.04 for the datasets considered and for the same threshold the value of $\frac{L_{avg}}{H_{avg}}$ ranges from 0.07 to 0.14. So, the value of $\frac{K_H}{K_L}$ ranges from $0.000009 * (0.0049)^{\alpha}$ to $0.019 * (0.0016)^{\alpha}$. Further, the value of α lies between 2 and 3. So, $\frac{K_H}{K_L}$ can be very small. We show the corresponding values in Table 2

Now we present our modifications to the above metrics, let us call them $CN1$, $AA1$ and $RA1$ where these stand for the modified similarity functions for CN, AA and RA respectively. Let $R = N(x) \cap N(y)$.

$$CN1(x,y) \; = \mid S(z) \mid \; where \; S(z) = \{z | z \in R \wedge degree(z) < T\} \tag{20}$$

$$AA1(x,y) \; = \sum_{z \in R \wedge degree(z) < T} \frac{1}{log(degree(z))} \tag{21}$$

$$RA1(x,y) = \sum_{z \in R \wedge degree(z) < T} \frac{1}{\sqrt{degree(z)}} + \sum_{z \in R \wedge degree(z) \geq T} \frac{1}{degree(z)^2} \tag{22}$$

Thus, in general we can write the score function as a combination of two monotonically non-increasing functions *low* and *high* , where low is applied on common neighbors having degree less than threshold and high is applied on common neighbors having degree greater than the threshold.

$$score(x,y) = \sum_{z \in R} (low(z) + high(z)) \tag{23}$$

Table 1. $low(z)$ and $high(z)$ for various schemes

Metric	low(z)	high(z)
CN1	1	0
AA1	$\frac{1}{log(degree(z))}$	0
RA1	$\frac{1}{\sqrt{degree(z)}}$	$\frac{1}{degree(z)^2}$

where z is the common neighbor of x and y. In our approach we use values for functions low and high as shown in table 1. It is clear from our approach that we have given less importance or zero weight to high degree nodes. Let z_1 and z_2 be two common neighbors of nodes x and y with degrees p and q respectively such that $z_1 \in L$ and $z_2 \in H$ then their contributions are:

- contribution of z_1 to RA is $\frac{1}{p}$ and for RA1 it is $\frac{1}{\sqrt{p}}$; further, $\frac{1}{\sqrt{p}} > \frac{1}{p}$
- contribution of z_2 to RA is $\frac{1}{q}$ and for RA1 it is $\frac{1}{q^2}$; further, $\frac{1}{q^2} < \frac{1}{q}$
- contribution of z_1 to CN is same as its contribution in CN1 and contribution of z_2 does not contribute to CN1 as it contributes in CN. Similar interpretation holds for AA and AA1.

So, common neighbors in L contribute more to RA1 than RA and those in H contribute less to RA1 than to RA.

Table 2. Details of various datasets

Dataset	$\mid V \mid$	$\mid E \mid$	T	L_{avg}	H_{avg}	n_L	n_H	$\frac{L_{avg}}{H_{avg}}$	$\frac{n_H}{n_L}$
GrQc	5241	28968	65	9	99	5168	73	0.09	0.014
HepTh	9875	51946	53	9	80	9792	83	0.1125	0.008
CondMat	23133	186878	45	13	92	22221	912	0.14	0.04
AstroPh	18771	396100	322	40	551	18714	57	0.07	0.003

4 Datasets and Experimental Methodology

For conducting our experiments we used collaboration graphs [6] which are undirected. In the case of collaboration graphs, the nodes represent the authors and the edge between two nodes x and y represents a collaboration between authors x and y. We conduct the experiments on the four datasets listed in table 2. We remove the self loops and nodes having degree one. The details of the datasets after preprocessing are given in Table 2. The datasets for papers from January 1993 to April 2003 are : AstroPh (Astro Physics), CondMat (Condense Matter Physics), GrQc (General Relativity and Quantum Cosmology), and HepTh (High Energy Physics - Theory).

We perform the link prediction on a static snapshot of the graph. We follow the approach to set up the data similar to the one explained in [2] and [5]. We take each of the graph datasets and randomly partition the graph into five parts by removing the edges randomly where each part has 20% of the edges. Now, we use one part as test data and the remaining 4 parts for training. That is we use a 5 fold cross validation. We repeat this five times each time taking a different part to be the test data; we report the average values.

5 Results and Discussions

On performing the experiments using the modified metrics described in section 3 we report the results in table 3 which indicate the accuracy on various datasets. From table 3 we observe that RA outperforms CN and AA. This result is consistent with the results shown in [2]. We can observe that our modified approach for each of the similarity functions CN, AA and RA performs better than the corresponding base metric as shown in boldface. We can observe that on the above datasets, CN1 performs almost as well as AA and AA1 performs as well as RA. Further, RA1 is the best. Also note that in some cases the accuracy has increased up to 12%.

Table 3. Percentage Classification Accuracy

Dataset	CN	CN1	AA	AA1	RA	RA1
GrQc	98.4	99.3	98.7	99.5	98.4	99.5
HepTh	78.1	85.4	90.4	93.8	92.2	94.5
CondMat	81.61	93.42	92.41	96.76	96.76	97.13
AstroPh	98.81	99.02	99.36	99.45	99.45	99.55

6 Conclusion

Based on experimentation, we observed that our approach performs better than the corresponding base similarity functions. RA1 performs the best among all the metrics. Thus, from the results we can conclude that our approximation of the state-of-the-art local neighborhood similarity functions performs better than the original similarity functions in terms of classification accuracy; further, it can decrease time when the contribution of high degree neighbors is ignored. We can conclude that high degree neighbors are not so useful in predicting new links. Thus, we can completely ignore or minimize the contributions of high degree nodes by making use of a suitable non-linear similarity function to weigh their contributions accordingly.

References

1. Nowell, L., Kleinberg, J.: The Link Prediction Problem for Social Networks. In: CIKM, pp. 556–559. ACM Press, New York (2003)
2. Zhou, T., Lu, L.: Link Prediction in Complex networks. J. Phys. A 390, 1150–1170 (2011)
3. Adamic, L., Adar, E.: Friends and neighbours on the Web. J. Social Networks 25, 211–230 (2003)
4. Zhou, T., Lu, L., Zhang, Y.-C.: Predicting Missing links via local information. J. Eur. Phys. B 71, 623–630 (2009)
5. Soundarajan, S., Hopcroft, J.: Using Community information to Improve the Precision of Link Prediction Methods. In: WWW, pp. 607–608. ACM Press, New York (2012)
6. Leskovec, J., Kleinberg, J., Faloutsos: Graph Evolution: Densification and shrinking Diameters. Technical report, arXiv.org (2007)

Optimal Pair of Coupling Function and STDP Window Function for Auto-associative Memory

Ryota Miyata, Keita Sato, and Toru Aonishi

Interdisciplinary Graduate School of Science and Engineering,
Tokyo Institute of Technology
4259–G5–17, Nagatsuta-cho, Midori-ku, Yokohama, Kanagawa, Japan
aonishi@dis.titech.ac.jp

Abstract. We verify whether the optimal pairs of coupling functions and spike-timing-dependent plasticity (STDP) window functions for executing the auto-associative memory algorithm derived from the viewpoint of hardware implementation are identical to those derived from the computational viewpoint by Lengyel et al. (2005). With a zero noise limit, we obtained the same relation between the coupling function and the STDP window function as that which Lengyel et al. obtained.

Keywords: coupling function, STDP, auto-associative memory.

1 Introduction

The hippocampus is an information processing system designed to address the task of storing memories and retrieving them. Marr [1] defined three levels at which such information processing systems must be understood completely as *computational*, *algorithmic*, and *implementational*. Contrasting the previous studies using mathematical models related to hippocampal memory with Marr's tri-level hypothesis [1], they can be divided broadly into two categories:

One is the top-down approach to bridge the computational and algorithmic levels. Lengyel et al. [2] derived an algorithm related to auto-associative memory function in the hippocampal area CA3 by treating auto-associative memory retrieval as a kind of Bayesian inference. Moreover, they developed a theory for the auto-associative memory algorithm that specifies an optimal relation between the synaptic plasticity rule for storing memories and the form of neural interactions for memory retrieval. Given the speculation that a coupling function is appropriate to formulate the neuronal interactions if memories are stored by spike-timing-dependent plasticity (STDP), they derived the relation between the coupling function $\Gamma(\phi_i - \phi_j)$ and the STDP window function $\Omega(\phi_i - \phi_j)$:

$$\Gamma(\phi_i - \phi_j) \propto \frac{\partial}{\partial \phi_i} \Omega(\phi_i - \phi_j), \tag{1}$$

where ϕ_i and ϕ_j respectively denote presynaptic and postsynaptic spike phases.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 265–272, 2013.

The other is the bottom-up approach to bridge the implementational and algorithmic levels. In our previous work [3], we derived an algorithm related to hetero-associative memory function in the hippocampal area CA1, by application of the reductive perturbation method [4] into a high-dimensional nonlinear dynamical system of neurons. Thereafter, we derived optimal pairs of coupling functions and STDP window functions for executing the hetero-associative memory algorithm.

Unfortunately, a relation between our algorithm derived from the bottom up and Lengyel's algorithm derived from the top down has not yet been reported. We now address this issue. In this paper, we verify whether the optimal pairs of coupling functions and STDP window functions for executing the auto-associative memory algorithm derived by our bottom-up approach are identical to Lengyel's results [2]. We can understand memories in the hippocampal CA3 at the three levels if we obtain results consistent with them.

Under the assumption of regular spiking and weak coupling, we first formulate an auto-associative memory model retrieving spike patterns as a phase oscillator model consisting of a coupling function and an STDP window function. Next, we analytically derive the mutual information between a stored spike pattern and the retrieval pattern, and use it to evaluate the memory retrieval performance. By maximizing the mutual information, we search for a set of optimal coupling functions under the same constraint of the STDP window function as Lengyel et al. [2]. The theoretically derived coupling functions are compared with the previous study. In the zero noise limit, we obtained the same relation between the coupling function and the STDP window function as that which Lengyel et al. obtained.

2 Methods

2.1 Minimum Model Functioning As an Auto-associative Memory

In line with the previous study [2], we assume that the hippocampal CA3 works as auto-associative memory retrieving a spatiotemporal spike pattern from a partial or noisy cue. We formulate a minimum model functioning as a temporal auto-associative memory as the recurrent network model presented in Fig. 1. The model consists of N oscillator neurons. For simplicity, these neurons have the same spiking period. Neuron i is connected to neuron j through a synapse with an efficacy (weight) J_{ij}. The theoretical derivations presented below assume all-to-all connectivity.

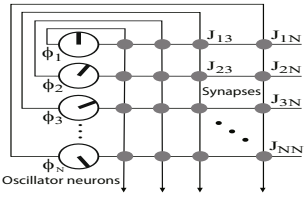


Fig. 1. Schematic diagram of a recurrent network with oscillator neurons. Neurons numbered $i = 1, \cdots, N$ are characterized by their phases $\phi_1, \cdots, \phi_N$, representing their individual spike timings. The angle of the radius line in the circle represents the phase. Neurons are fully connected by N^2 synaptic connections.

For mathematical tractability and simplicity, we assume that the timescale of synapse dynamics in the memory storage process is far different from that of network dynamics in the memory retrieval process. Under such an assumption of timescale separation, the storage process and retrieval process are separable from one another.

Synapse Dynamics in the Storage Process. In the storage process, we treat θ_j as the spike phase of presynaptic neuron j $(= 1, \cdots, N)$, and θ_i as that of postsynaptic neuron i $(= 1, \cdots, N)$. Also, J_{ij} denotes the synaptic weight between presynaptic neuron j and postsynaptic neuron i. Memory storage occurs as a result of synaptic weight change. The amount of synaptic weight change, ΔJ_{ij}, is determined according to the following synaptic plasticity rule as

$$\Delta J_{ij} = \frac{1}{N} \Omega(\theta_i - \theta_j). \tag{2}$$

Therein, $\Omega(\cdot)$ is the STDP window function, for which the relative phase (timing) of presynaptic and postsynaptic spikes determines the sign and extent of synaptic weight change. When storing more than one pattern $(\mu = 1, \cdots, p)$, we also make a simplifying assumption that synaptic plasticity is additive across memories.

$$J_{ij} = \frac{1}{N} J_0 + \frac{1}{N} \sum_{\mu=1}^{p} \Omega(\theta_i^\mu - \theta_j^\mu), \tag{3}$$

Therein, $\frac{1}{N} J_0$ (> 0) is the initial synaptic weight, which avoids negative values of J_{ij}. This plasticity rule is similar to that described in the previous study [2]. The difference from the previous study is the scaling of the synaptic weight with the number of neurons, N. This scaling is necessary for derivation of the order parameter $m_{k,l}^\mu$, which measured the overlap between the μ-th memory pattern $\boldsymbol{\theta}^\mu$ and the retrieval pattern $\boldsymbol{\phi}$, as we discuss later. Because the magnitude of $\Omega(\cdot)$ is arbitrary, there is no loss of generality because of the scaling in Eqs. (2) and (3). Each element of the μ-th memory pattern θ_i^μ is assigned to an independent random number in $[0, 2\pi)$ with a uniform probability, $P(\theta_i^\mu) = \frac{1}{2\pi}$. Consequently, these memory patterns are not mutually correlated.

Network Dynamics in the Retrieval Process. In the retrieval process, we treat ϕ_j as the spike phase of presynaptic neuron j, and ϕ_i as that of postsynaptic neuron i. When the neurons have the same spiking period (i.e., regular spiking) and the synaptic weight J_{ij} is sufficiently small (i.e., weak coupling), the retrieval dynamics can be expressed by the Langevin phase equation (LPE)[3,5]:

$$\frac{d\phi_i}{dt} = \sum_{j=1}^{N} J_{ij} \Gamma(\phi_i - \phi_j) + \sigma s_i(t). \tag{4}$$

$\Gamma(\cdot)$ is called the coupling function, indicating the effect of presynaptic spike phase ϕ_j on the postsynaptic one ϕ_i. $s_i(t)$ is white noise satisfying $< s_i(t) >= 0$, $< s_i(t) s_j(t') >= 2\delta_{ij}\delta(t - t')$. σ represents the noise intensity.

Here, we expand the STDP window function and the coupling function into their Fourier series:

$$\Omega(\theta_i^\mu - \theta_j^\mu) = \sum_{k=-\infty}^{\infty} a_k \exp(ik(\theta_i^\mu - \theta_j^\mu)), \quad a_k = \frac{1}{2\pi}\int_{-\infty}^{\infty} d\theta \Omega(\theta)\exp(-ik\theta), \quad (5)$$

$$\Gamma(\phi_i - \phi_j) = \sum_{l=-\infty}^{\infty} b_l \exp(il(\phi_i - \phi_j)), \quad b_l = \frac{1}{2\pi}\int_{-\infty}^{\infty} d\phi \Gamma(\phi)\exp(-il\phi), \quad (6)$$

where a_k and b_l respectively denote the Fourier coefficients of the STDP window function and the coupling function (k and l denote frequency components). $\Omega(\cdot)$ and $\Gamma(\cdot)$, which are real-valued functions, satisfy $a_{-k} = a_k^*$ and $b_{-l} = b_l^*$ (superscript $*$ denotes the complex conjugate), respectively. The initial synaptic weight J_0 can be involved in the DC component a_0 without loss of generality. Here, we define A_k and ζ_k respectively as the amplitude and the phase of a_k, ($a_k = A_k \exp(i\zeta_k)$). B_l and χ_l represent the amplitude and the phase of b_l ($b_l = B_l \exp(i\chi_l)$). They satisfy $A_{-k} = A_k$, $B_{-l} = B_l$, $\zeta_{-k} = -\zeta_k$, $\chi_{-l} = -\chi_l$.

The order parameter $m_{k,l}^\mu$, the overlap between the μ-th memory pattern $\boldsymbol{\theta}^\mu$ and the retrieval pattern $\boldsymbol{\phi}$ in each frequency component, is defined as

$$m_{k,l}^\mu = \frac{1}{N}\sum_{j=1}^{N} \exp(i(k\theta_j^\mu - l\phi_j)), \quad (7)$$

All the neurons share the same order parameter: $m_{k,l}^\mu$.

Using a_k, b_l and $m_{k,l}^\mu$, the LPE (4) can be transformed into

$$\frac{d\phi_i}{dt} = \sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\sum_{\mu=1}^{p} a_k^* b_l m_{k,l}^\mu \exp(i(l\phi_i - k\theta_i^\mu)) + \sigma s_i(t). \quad (8)$$

The neurons share the same $m_{k,l}^\mu$ and are driven by independent noise. Therefore, the N neurons can be regarded as statistically mutually independent and as having the same statistical characteristics.

2.2 Equilibrium Phase Distribution When Storing a Finite Number of Patterns

We here consider the case in which the number of stored patterns p is finite in the thermodynamic limit as $N \to \infty$ (i.e., $p << N$). The first memory pattern is to be retrieved ($\boldsymbol{\phi} \approx \boldsymbol{\theta}^1$). As described above, the memory patterns $\boldsymbol{\theta}^\mu$ ($\mu = 1, \cdots, p$) are not mutually correlated.

Each overlap $m_{k,l}^\mu$ is calculable as follows. The average overlap with $\mu = 1$ between two components with the same frequency ($k = l$) is

$$\left\langle m_{l,l}^1 \right\rangle = \frac{1}{N}\sum_{j=1}^{N} \left\langle \exp(il(\theta_j^1 - \phi_j)) \right\rangle = O(1). \quad (9)$$

However, the average overlap with $\mu = 1$ between the other components ($k \neq l$) is $\left\langle m_{k,l}^1 \right\rangle = 0$ and the deviation is $O(1/\sqrt{N})$. Moreover, for any k and l, the

overlap between the retrieval pattern $\boldsymbol{\phi}$ and a memory pattern other than the first ($\mu \geq 2$) is on average $\left\langle m^{\mu}_{k,l} \right\rangle = 0$, and the deviation is $O(1/\sqrt{N})$. In the case of a finite p, the number of terms with $\mu \geq 2$ in Eq. (8) is finite. Moreover, the Fourier coefficients a_l and b_l rapidly approach zero as l increases. Consequently, contributions of the terms except those with $\mu = 1$ and $k = l$ to Eq. (8) are negligible in the limit $N \to \infty$. The phase dynamics can be rewritten as

$$\frac{d\phi_i}{dt} = 2\sum_{l=1}^{\infty} \frac{A_l B_l |m^1_{l,l}|}{l} \sin\big(l(\phi_i - \theta^1_i) - \zeta_l + \chi_l + \angle m^1_{l,l} - \frac{\pi}{2})\big) + \sigma s_i(t). \quad (10)$$

The term consisting of DC components a_0 and b_0 in Eq. (8) is a constant. We can safely neglect this term because the constant term can be involved in the natural frequency ω without loss of generality. Equation (10) shows that the statistical properties of the auto-associative memory model in the case of finite loading ($p << N$) and those in the simplest case of just one pattern to be stored ($p = 1$) are identical.

From Eq. (10), we obtain the equilibrium phase distribution of each neuron:

$$P(\phi_i|\theta^1_i) = \frac{1}{Z_{\mathrm{NF}}} \exp\big(\frac{2}{\sigma^2} \sum_{l=1}^{\infty} \frac{A_l B_l |m^1_{l,l}|}{l} \cos(l(\phi_i - \theta^1_i) - \zeta_l + \chi_l + \angle m^1_{l,l} - \frac{\pi}{2})\big), \quad (11)$$

$$Z_{\mathrm{NF}} = \int_0^{2\pi} d\phi \exp\big(\frac{2}{\sigma^2} \sum_{l=1}^{\infty} \frac{A_l B_l |m^1_{l,l}|}{l} \cos(l\phi_i - \zeta_l + \chi_l + \angle m^1_{l,l} - \frac{\pi}{2})\big), \quad (12)$$

where Z_{NF} is the normalizing factor. In a self-consistent manner [5], we can obtain the order parameter equation, which denotes the overlap at equilibrium:

$$m^1_{l,l} = \frac{1}{N} \sum_{i=1}^{N} \exp(il(\theta^1_i - \phi_i)) = \int_0^{2\pi} d\phi_i P(\phi_i|\theta^1_i) \exp(-il\phi_i). \quad (13)$$

2.3 Mutual Information per Neuron

Mutual information, which measures how related two random variables are, is nonnegative and takes 0 only if these variables are independent.

When retrieving the first memory pattern $\boldsymbol{\theta}^1$, the mutual information of the retrieval pattern $\boldsymbol{\phi}$ to $\boldsymbol{\theta}^1$ per neuron, $\frac{1}{N} H(\boldsymbol{\phi}; \boldsymbol{\theta}^1)$, is given as

$$\frac{1}{N} H(\boldsymbol{\phi}; \boldsymbol{\theta}^1) = \frac{1}{N} H(\boldsymbol{\phi}) - \frac{1}{N} H(\boldsymbol{\phi}|\boldsymbol{\theta}^1). \quad (14)$$

Here, $\frac{1}{N} H(\boldsymbol{\phi})$ is the entropy of $\boldsymbol{\phi}$ per neuron, which measures the uncertainty associated with $\boldsymbol{\phi}$. $\frac{1}{N} H(\boldsymbol{\phi}|\boldsymbol{\theta}^1)$ is the conditional entropy of $\boldsymbol{\phi}$ given $\boldsymbol{\theta}^1$, which quantifies the remaining uncertainty of $\boldsymbol{\phi}$ given that $\boldsymbol{\theta}^1$ is known. Because each neuron is statistically independent and has the same statistical characteristics as those described above, $\frac{1}{N} H(\boldsymbol{\phi})$ and $\frac{1}{N} H(\boldsymbol{\phi}|\boldsymbol{\theta}^1)$ can be written simply as follows.

$$\frac{1}{N} H(\boldsymbol{\phi}) = -\frac{1}{N} \sum_{i=1}^{N} \int_0^{2\pi} d\phi_i \int_0^{2\pi} d\theta^1_i P(\phi_i|\theta^1_i) P(\theta^1_i) \ln \int_0^{2\pi} d\theta_i P(\phi_i|\theta^1_i) P(\theta^1_i) = \ln 2\pi, \quad (15)$$

$$\frac{1}{N}H(\boldsymbol{\phi}|\boldsymbol{\theta}^1) = -\frac{1}{N}\sum_{i=1}^{N}\int_0^{2\pi} d\theta_i^1 P(\theta_i^1)\int_0^{2\pi} d\phi_i P(\phi_i|\theta_i^1)\ln P(\phi_i|\theta_i^1)$$

$$= -\frac{2}{\sigma^2 Z_{\mathrm{NF}}}\int_0^{2\pi} d\phi \sum_{l=1}^{\infty}\frac{A_l B_l |m_{l,l}^1|}{l}\cos(l\phi - \zeta_l + \chi_l + \angle m_{l,l}^1 - \frac{\pi}{2})$$

$$\times \exp\big(\frac{2}{\sigma^2}\sum_{l=1}^{\infty}\frac{A_l B_l |m_{l,l}^1|}{l}\cos(l\phi - \zeta_l + \chi_l + \angle m_{l,l}^1 - \frac{\pi}{2})\big) + \ln Z_{\mathrm{NF}}. \quad (16)$$

Because $\frac{1}{N}H(\boldsymbol{\phi})$ in Eq. (15) is a constant, maximization of the mutual information $\frac{1}{N}H(\boldsymbol{\phi};\boldsymbol{\theta}^1)$ in Eq. (14) is identical to minimization of the conditional entropy $\frac{1}{N}H(\boldsymbol{\phi}|\boldsymbol{\theta}^1)$ in Eq. (16).

3 Results

3.1 Retrieval Performance of the Network with Typical Parameters

We conducted numerical simulations on the auto-associative memory model with typical parameters. Then we compared the numerical results with theoretical predictions. Here, we use the auto-associative memory model endowed with the STDP window function (Fig. 2(a)) and the coupling function (Fig. 2(b)), which are used in the previous study [2]. In the following numerical simulations, three random phase patterns $\boldsymbol{\theta}^\mu$ $(\mu = 1, 2, 3)$ were stored; $\boldsymbol{\theta}^1$ was retrieved.

We verified the effect of intrinsic noise on the retrieval performance of this model. Figure 3 depicts the absolute value of overlap at equilibrium, $|m_{l,l}^1|$ $(l = 1, \cdots, 4)$, vs. the noise intensity σ. In this figure, the LPE (8) was solved numerically using the Euler method. The values of $|m_{l,l}^1|$ calculated with Eq. (7) were compared with theoretical predictions obtained by Eq. (13). As depicted in Fig. 3, the numerical results coincide with the theoretical values for all frequency components l. The retrieval pattern $\boldsymbol{\phi}$ has an appreciable overlap with $\boldsymbol{\theta}^1$, i.e., $|m_{l,l}^1| \approx O(1)$ if the noise intensity σ is less than 0.1. However, if $\sigma > 0.1$, then $|m_{l,l}^1|$ converges to zero. Consequently, the network with the parameters used in the previous study [2] functions as an auto-associative memory within $\sigma < 0.1$.

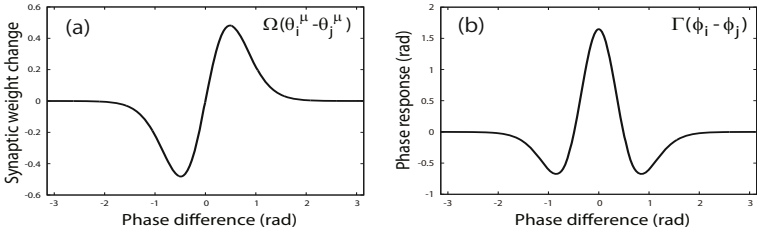


Fig. 2. (a) STDP window function used in the previous study [2], $\Omega(\theta_i^\mu - \theta_j^\mu)$, which is a continuous fit to the data of hippocampal neurons [6]. (b) Coupling function derived from the previous study [2], $\Gamma(\phi_i - \phi_j)$, which is optimally matched to the STDP (a).

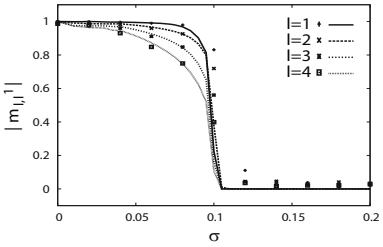


Fig. 3. Absolute values of overlaps at equilibrium, $|m^1_{l,l}|$ $(l = 1, \cdots, 4)$, as a function of noise intensity σ. We use the same STDP (Fig. 2(a)) and coupling function (Fig. 2(b)) as Lengyel et al. [2]. Curve lines are theoretical results obtained from Eq. (13). The points are obtained from numerical simulations using Eqs. (7, 8). In this simulation, $N = 1000$ and $p = 3$.

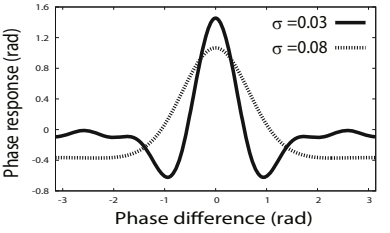


Fig. 4. Examples of coupling functions optimally matched to the hippocampal STDP window function (Fig. 2(a)), $\Gamma(\phi_i - \phi_j)$, which are derived by maximizing the mutual information (14). The solid line is an example of coupling functions when $\sigma = 0.03$. The point is one when $\sigma = 0.08$.

3.2 Coupling Functions Optimally Matched to the Hippocampal STDP Window Function

We searched for coupling functions that were optimally matched to the hippocampal STDP window function presented in Fig. 2(a). Substituting the Fourier coefficients of the hippocampal STDP into a_l of the mutual information (14), and under the constraint of the STDP, we searched for b_l $(l = 1, \cdots, 4)$, the Fourier coefficients of optimal coupling function to maximize the mutual information. Because the mutual information (14) increases monotonically as B_l (the amplitude of b_l) increases, we imposed the following constraint condition on the power of coupling function:

$$\sum_{l=1}^{4} B_l^2 \leq \text{Const.} \tag{17}$$

Referring to the value of the power of coupling function in the previous study [2], we set Const. $= 0.12$. To solve this optimization problem, we used the FMINCON function in the Matlab Optimization Toolbox.

Figure 4 shows examples of the coupling function derived by mutual information maximization. A set of optimal coupling functions forms a continuous set equipped with a ring topology. As portrayed in Fig. 4, when the noise intensity is small ($\sigma = 0.03$), the theoretically derived coupling function almost conforms to that derived by Lengyel et al. [2]. However, for a large noise intensity ($\sigma = 0.08$), the derived coupling function has a shape like a Gaussian function.

We verified the retrieval performance of the network model with the optimal coupling functions including Fig. 4 and the hippocampal STDP window function (Fig. 2(a)) in numerical simulations, and confirmed that the model works as an auto-associative memory (results not shown).

4 Discussion

We verify whether the optimal pairs of coupling functions and STDP window functions for executing the auto-associative memory algorithm derived from the viewpoint of hardware implementation coincide with those derived from the computational viewpoint by Lengyel et al. [2]. Using mutual information maximization, we derived coupling functions that were optimally matched to the classical form of the hippocampal STDP window function, which was the same constraint as that used in [2]. When the noise intensity was sufficiently small ($\sigma = 0.03$), the derived coupling function closely resembled that derived by Lengyel et al.

To speculate on the outcome, we here derive the limit of the mutual information (14) when σ goes to zero. By evaluating the integrals in Eqs. (12) and (16) with the saddle point method, we obtain the mutual information in the limit of zero noise as $\frac{1}{N}H(\boldsymbol{\phi};\boldsymbol{\theta}^1) = \ln(\sigma) + \frac{1}{2}\ln\sum_{l=1}^{\infty}(lA_lB_l) + \text{Const.}$ We can evaluate the maximum value of this function using the following Cauchy–Schwarz inequality: $\left(\sum_{l=1}^{\infty}(lA_lB_l)\right)^2 \leq \sum_{l=1}^{\infty}(lA_l)^2\sum_{l=1}^{\infty}(B_l)^2$. We have equality if and only if

$$B_l = lA_l, \quad l = 1, 2, \cdots. \tag{18}$$

Equation (18) shows that equality in the inequality holds if the coupling function is determined by the derivative of the STDP window function. Therefore, in the zero noise limit, we obtained the same relation between the coupling function and the STDP window function as that obtained by Lengyel et al. (i.e., Eq. (1)).

Finally, we express our gratitude to Japanese Neural Network Society (JNNS) for supporting English proofreading.

References

1. Marr, D.: Vision. Freeman (1982)
2. Lengyel, M., Kwag, J., Paulsen, O., Dayan, P.: Matching storage and recall: hippocampal spike timing-dependent plasticity and phase response curves. Nat. Neurosci. 8, 1677–1683 (2005)
3. Miyata, R., Ota, K., Aonishi, T.: Optimal design for hetero-associative memory: hippocampal CA1 phase response curve and spike-timing-dependent plasticity. Accepted for PLOS ONE (2013), doi:10.1371/journal.pone.0077395
4. Kuramoto, Y.: Chemical Oscillations, Waves, and Turbulence. Springer (1984)
5. Aonishi, T., Kurata, K., Okada, M.: A statistical mechanics of an oscillator associative memory with scattered natural frequencies. Phys. Rev. Lett. 82, 2800–2803 (1999)
6. Bi, G.Q., Poo, M.M.: Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. J. Neurosci. 18, 10464–10472 (1998)

Using Discriminative Phrases for Text Categorization

Sreya Dey and M. Narasimha Murty

Indian Institute of Science, Bangalore, India
{sreya,mnm}@csa.iisc.ernet.in
http://www.csa.iisc.ernet.in

Abstract. This work aims at finding discriminative multi-word features or phrases from a given set of text documents. We use a modified Shapley value measure to select the list of discriminative features. The feature selection algorithm is employed on 20-newsgroup and Wikipedia datasets, along with several existing classifiers. Based on the results obtained, we show that adding phrases to the feature list can improve classification performance in comparison to using words alone; further, the improvement varies depending upon the separability of the classes.

Keywords: Text classification, Feature Selection, Discriminative Phrases.

1 Introduction

Classification of text documents is an important task in today's scenario of ever-increasing volume of information. Considerable work has been done on designing classification methods for text documents. Most of these methods represent each document as a vector of its term frequencies. This Bag-of-Words (BOW) model has proved to be quite effective in the past. However, including phrases in the feature list can often increase the semantic information contained in the representation of the documents. Therefore, the idea of terms in the context of text classification has been extended from single words to phrases.

Our aim is to find such discriminative phrases and use them for classification. To find such discriminative features, we use the concept of Shapley value. We show that the game defined in this case is concave. Also, we employ statistical t-test to show that both the terms in most discriminative bigrams come from the the same zone in the Zipf's curve.

2 Related Work

The challenges involved in designing a *good* classifier are mainly twofold : First, the classification performance on the test documents should be good and two, the model constructed should be simple. Dimensionality reduction is one of the measures towards constructing a simple model for classification. Mutual Information[1] has been used to select features for text classification previously in [2], whereas works like [3] employ frequency-based feature selection.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 273–280, 2013.

Previously, work has been done to determine whether the use of phrases can improve the classification performance, with both positive and negative results. Caropreso et. al.[4] define a scoring scheme for terms and uses top ranked n-grams for classification. Their results show that often using phrases along with words bring down the classification accuracy. Tan et. al. in [3] have used most frequent words to generate phrases, from which top few are selected.

The solution concept of Shapley value has been used previously for feature selection. In [5], the backward elimination version of the Contribution Selection Algorithm is used, where in every iteration, the features with the lowest Shapley values are eliminated. In [6] and [7], Sun et. al. have used feature interdependency measures as the characteristic function and selected the most discriminative features using Shapley value.

3 Our Work

3.1 Analysis of Bigrams

From Zipf's Law, we know that the frequency of a word in a document collection is inversely proportional to its rank. Now we try to relate the frequency of a bigram to the ranks of its constituent words. We use a the two-tailed, one sample t-test to show that most significant bigrams usually comprise of words from the same frequency range. We rank the words in a class by their collection frequency and divide them into the following zones : Ranks 1-10 (Zone 1), 11-100 (Zone 2), 101-1000 (Zone 3), and so on. Now, we consider all possible bigrams and rank them by their MI with respect to that class. Now, we take the top 100 bigrams and for each bigram, we find the difference(d) between the zones in which the first and the second terms of the bigram lie. Our null hypothesis is that the mean of the values of d is equal to 0. We use the classes 'sci.crypto' and 'comp.graphics' for the test, considering the bigrams from the class 'sci.crypto'.

Null Hypothesis : $H_0 : \mu_0 = 0$
Alternative Hypothesis : $H_1 : \mu_0 \neq 0$
Sample size(n) = 100
Number of degrees of freedom = 100-1 = 99
Sample mean($\bar{x}$) = -0.02
Sample standard deviation(s) = 1.0148
Test statistic(t) = $\frac{\bar{x}-\mu_0}{s/\sqrt{n}}$ = -0.1970

From the value of t, the value of p is calculated to be 0.8442. Since this value is greater than 0.05, the null hypothesis is not rejected. So, we can say that the distribution of d values follow a t-distribution with mean 0 i.e. *most discriminative bigrams consist of words from the same zone in the rank spectrum.* Now, if we rank the bigrams in term of their collection frequencies and plot their ranks versus their MI values, we get the plot in Figure 1. This shows that the discriminative bigrams are generally more frequent. Similar results are observed for other classes as well.

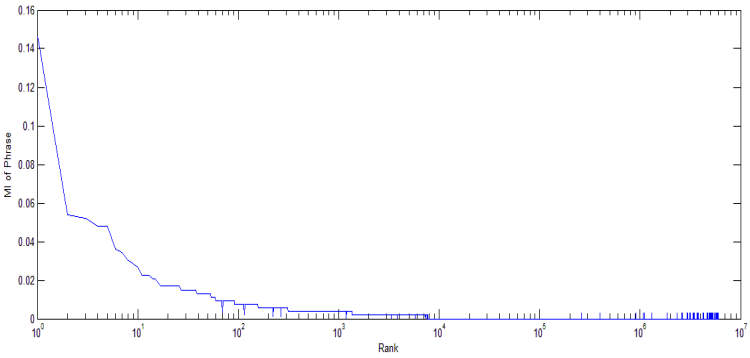


Fig. 1. MI(y-axis) versus rank(x-axis) of all bigrams

3.2 Algorithm for Feature Selection

Our feature selection algorithm uses the concept of Shapley value in a transferable utility(TU) game to assign a significance measure to each feature. The algorithm is as follows:

1. For each class, we select *wdim* words having the highest MI with respect to that class. The value of *wdim* is chosen such that at least one of the top *wdim* words appears in every document in the class. The reason for this is explained later. For binary classification, these two sets are the same. So in that case, we arrange the words in descending order of their MI and assign each of them to the class where the word is ranked higher in terms of frequency till we obtain *wdim* words for each class.
2. Now, for each class, we consider the possible $(wdim)^2$ bigrams that can be constructed using these *wdim* words. Thus we have a set N consisting of $wdim + (wdim)^2$ features from each class. Now we consider this set N as the set of players of a cooperative game. We define the characteristic function for a set of features C, $v(C)$, as the total fraction of documents in which at least one of these features is present, which we also call the *Span* of that set. The reason behind this choice is that phrases are generally rarer than words. Also, we have seen previously that frequent phrases are generally more discriminative. So we aim to find the set of features that contribute the most to covering the documents in a class. We ensured previously that the *Span* of the entire set of features is 1. We calculate the Shapley value of the coalition using the following equation and take the top *tdim* features having the highest values.The values of parameters *wdim* and *tdim* are varied and the optimal values are selected.

$$\phi_i(N, v) = \frac{1}{n!} \sum_{p \in P} [v({C_i}^p \cup \{i\}) - v({C_i}^p)] \tag{1}$$

where $i \in N$, $n = |N|$ and P is the set of all possible permutations of the set N and ${C_i}^p$ represents the set of players that come before i in the permutation p.

Approximate calculation of Shapley value. To compute the Shapley value, we have to consider $n!$ possible permutations of the n features. However, it has been shown in [8] that if p (where $p \ll n$) randomly selected permutations are chosen from the $n!$ possible permutations, the resulting error between the actual and calculated Shapley value is bounded. We vary parameter p while applying the algorithm to the data sets and see that the classification accuracy does not increase by changing the number of permutations beyond a certain threshold.

3.3 Nature of the Game

We shall now prove that the resulting game is concave. Let us consider two sets of features P and Q. We previously defined the *Span* of a set as the fraction of documents where at least one of the features in the set occur. To prove that the game is concave, we need to show that $Span(P) + Span(Q) \geq Span(P \cap Q) + Span(P \cup Q)$. Let us represent the sets $P - Q$, $P \cap Q$ and $Q - P$ by A, B and C respectively. If at least one phrase in a set occurs in a document, we say that the phrase set *spans* the document. In Figure 2, the three circles represent the documents spanned by sets A, B and C. The number of documents in each region of the Venn diagram is denoted by a, b, c, d, e, f, g and h. Let the total number of documents be s, where $s = a + b + c + d + e + f + g + h$.

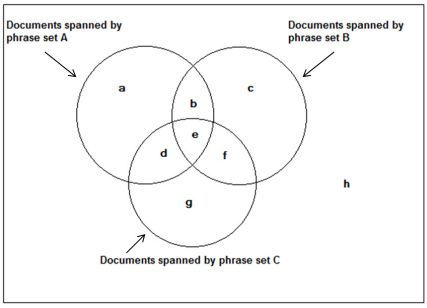


Fig. 2. Documents spanned by 3 phrase sets A, B and C

We can see that,
$Span(P) = (a + b + c + d + e + f)/s$
$Span(Q) = (b + c + d + e + f + g)/s$
$Span(P \cap Q) = (b + c + e + f)/s$
$Span(P \cup Q) = (a + b + c + d + e + f + g)/s$
$[\ Span(P) + Span(Q)\ ]$ - $[\ Span(P \cap Q) + Span(P \cup Q)\ ] = d/s$
Since, $d \geq 0$, $Span(P) + Span(Q) \geq Span(P \cap Q) + Span(P \cup Q)$.

Hence the game is concave. This means that the gain obtained in terms of document coverage by adding a feature to a set of features will be less than that obtained by adding it to a subset of that set.

4 Experimental Results

4.1 Datasets and Classifiers Used

Four classes from the 20-newsgroup dataset (www.cs.umb.edu/ smimarog/ textmining) after removal of stop words are used in this work. We have also used Wikipedia web pages from three categories. Details of both datasets are given in Table 1. For classification, we use Libsvm, k-NN and Decision Tree (Rapidminer).

Table 1. Classes used from 20-newsgroup dataset

Dataset	Class	Number of Training Documents	Number of Test Documents
20-newsgroup	sci.crypto	595	396
	rec.sport.hockey	600	399
	alt.atheism	480	319
	comp.graphics	584	389
Wikipedia	Data Mining	54	37
	Machine Learning	77	51
	Computer Graphics	130	86

4.2 Results

We compare the classification accuracies using (a) all features in the collection having a document frequency higher than a certain threshold, (b) top *wdim* words from each class, ranked using their MI and (c) top *tdim* features from each class selected using the above algorithm using Shapley value. The value of the parameters chosen in all the following cases are as follows: $wdim = 10$, $tdim = 10$, $p = 1000$, Frequency Threshold = 10. Table 2 shows the classification accuracies obtained for different pairs of classes and their corresponding Bhattacharyya Coeffiecients[9]. Figure 3 shows the effect of varying the number of random permutations p on the classification accuracies obtained using SVM classifier for each pair of classes.

4.3 Interpretation

The following remarks can be made from the results observed:

1. Most discriminative bigrams are frequent and consist of words which belong to the same frequency range.
2. The feature selection algorithm using Shapley value improves the classification accuracy in almost all cases, as compared to the baseline accuracy without feature selection. In most cases, there is an improvement in classification accuracy as compared to using words only.

Table 2. Classification accuracies obtained for different data and classifiers

Class 1(C_1)	Class 2(C_2)	BC(C_1,C_2)	Features Used	Accuracy (SVM)	Accuracy (Decision Tree)	Accuracy (k-NN)
Machine Learning	Data Mining	0.3611	All features	87.5	77.63	68.04
			Top *wdim* words	92.04	87.21	93.15
			Top *tdim* features	**93.18**	**88.58**	**93.61**
Machine Learning	Computer Graphics	0.0	All features	99.27	97.67	95.35
			Top *wdim* words	**100**	98.84	99.71
			Top *tdim* features	**100**	**100**	**100**
Data Mining	Computer Graphics	0.0	All features	98.54	97.72	61.24
			Top *wdim* words	**100**	**100**	99.67
			Top *tdim* features	**100**	**100**	**100**
sci.crypto	comp.graphics	0.201	All features	92.23	90.73	**83.25**
			Top *wdim* words	94.65	**92.41**	82.38
			Top *tdim* features	**94.90**	**92.41**	82.38
rec.sports.hockey	alt.atheism	0.1363	All features	**94.84**	**88.43**	81.87
			Top *wdim* words	94.71	87.99	**89.32**
			Top *tdim* features	**94.84**	87.99	88.93
sci.crypto	alt.atheism	0.0396	All features	91.05	94.56	83.54
			Top *wdim* words	**98.74**	**97.71**	**98.49**
			Top *tdim* features	**98.74**	**97.71**	**98.49**
sci.crypto	rec.sports.hockey	0.0807	All features	93.08	94.07	83.02
			Top *wdim* words	**97.36**	**94.12**	**96.23**
			Top *tdim* features	**97.36**	**94.12**	**96.23**

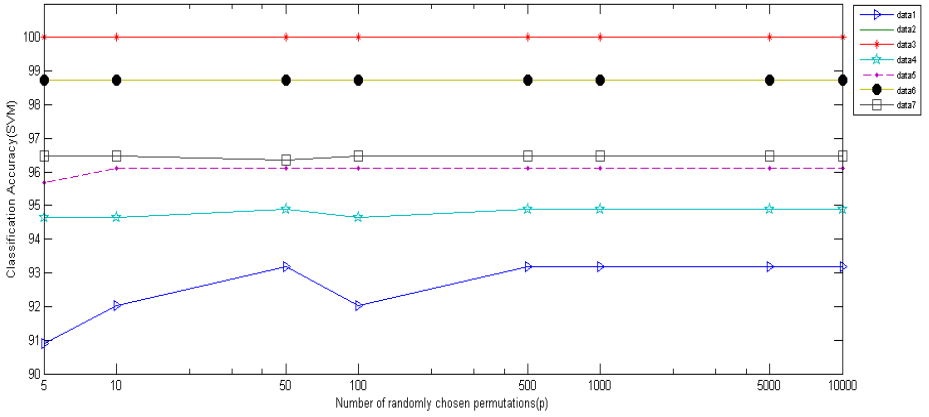


Fig. 3. Number of random permutations p versus classification accuracy

3. This improvement is greater for classes which have a higher overlap among them. For example, we can intuitively say that the classes 'Machine Learning' and 'Data Mining' will have a greater overlap in terms of common features as compared to the classes 'Cryptography' and 'Computer Graphics'. This can also be shown by comparing the Bhattacharyya Coefficient(BC) between these two pairs of classes, which ranges from 0 (completely separated classes) to 1(completely overlapping classes). We can see that the improvement obtained by adding phrases is greater for classes which have higher Bhattacharyya coefficient. This can be explained as follows: In classes with higher overlap, the number of common features is higher. In such cases, adding discriminative phrases can improve the classification accuracy (Table 2). However, in case of classes with lower overlap, using only words is almost as good as incorporating phrases. In some cases, where the classes are very well separated, adding phrases can even bring down the classification accuracy.
4. In all cases, the classification accuracy stabilizes at or below 500 random permutations. The total number of features is $(wdim + (wdim)^2) * 2$. Therefore, for a value of $wdim = 10$, there are 220! possible permutations. Therefore, we can say that the approximation is quite good and computationally efficient.
5. SVM performs much better than Decision Tree as well as k-NN in terms of classification accuracy, especially when the number of features is high. Also, SVM as well as k-NN outperforms Decision Tree in terms of time taken for training the classifier.

5 Conclusions and Future Work

From the results obtained, we can conclude that the effectiveness of adding bigrams to the feature list depends on the dataset. In case of classes which are well separated, it is seen that there is not much incentive to be gained from including phrases in the feature list. However, in case of overlapping classes, bigrams can provide more discriminative information as compared to words alone. Our approach could be extended to phrases of length greater than 2. But, such phrases are rarer and much less likely to improve the classification performance. Also, the game involved in feature selection is concave. Hence we cannot guarantee that the solution is in the core. One future direction could be to modify the characteristic function such that the resulting game is convex.

References

1. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
2. Bekkerman, R., Allan, J.: Using Bigrams in Text Categorization. Technical report IR-408, Center for Intelligent Information Retrieval, UMass Amherst (2004)
3. Tan, C.M., Wang, Y.F., Lee, C.D.: The use of bigrams to enhance text categorization. Information Processing and Management 38, 529–546 (2002)

4. Caropreso, M.F., Matwin, S., Sebastiani, F.: A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In: Text Databases and Document Management: Theory and Practice, pp. 78–102. Idea Group Publishing, Hershey (2001)
5. Cohen, S., Dror, G., Ruppin, E.: Feature selection via coalitional game theory. Neural Computation 19, 1939–1961 (2007)
6. Sun, X., Liu, Y., Li, J., Zhu, J., Chen, H., Liu, X.: Feature evaluation and selection with cooperative game theory. Pattern Recognition 45, 2992–3002 (2012)
7. Sun, X., Liu, Y., Li, J., Zhu, J., Liu, X., Chen, H.: Using cooperative game theory to optimize the feature selection problem. Neurocomputing 97, 86–93 (2012)
8. Garg, V.K., Narahari, Y., Murty, M.N.: Novel biobjective clustering (bigc) based on cooperative game theory. IEEE Transactions on Knowledge and Data Engineering 25, 1070–1082 (2013)
9. Bhattacharyya, A.K.: On a measure of divergence between two statistical populations defined by their probability distribution. Bulletin of the Calcutta Mathematical Society 35, 99–110 (1943)

A Random Block Coordinate Descent Method for Multi-label Support Vector Machine

Jianhua Xu*

School of Computer Science and Technology,
Nanjing Normal University, Nanjing, Jiangsu 210023, China
xujianhua@njnu.edu.cn

Abstract. Multi-label support vector machine (Rank-SVM) is an effective algorithm for multi-label classification, which is formulated as a quadratic programming problem with q equality constraints and lots of box constraints for a q-class multi-label data set. So far, Rank-SVM is solved by Frank-Wolfe method (FWM), where a large-scale linear programming problem needs to be dealt with at each iteration. In this paper, we propose a random block coordinate descent method (RBCDM) for Rank-SVM, in which a small-scale quadratic programming problem with at least $(q+1)$ variables randomly is solved at each iteration. Experiments on three data sets illustrate that our RBCDM runs much faster than FWM for Rank-SVM, and Rank-SVM is a powerful candidate for multi-label classification.

Keywords: Multi-label classification, support vector machine, Frank-Wolfe method, block coordinate descent method.

1 Introduction

Traditional supervised classification deals with problems in which one instance is only associated with a single class label and thus the classes are mutually exclusive. However, in many real world applications, one instance possibly belongs to several labels at the same time, e.g., a sunset image could be annotated by sun, sky and mountain simultaneously [1]. Such a classification issue is referred to as multi-label classification and has been attracted a lot of attention in the recent decade. So far, a variety of multi-label methods have been proposed and validated [2–4], among which multi-label support vector machine (Rank-SVM) [5, 6] is one of the most famous methods.

Let a finite set of q class labels be $L = \{1, 2, ..., q\}$ and its all possible subsets be 2^L. We denote a training set of the size l drawn identically and independently from an unknown probability distribution on $R^d \times 2^L$ by

$$\{(\boldsymbol{x}_1, L_1), ..., (\boldsymbol{x}_i, L_i), ..., (\boldsymbol{x}_l, L_l)\}, \tag{1}$$

* This study was supported by the Natural Science Foundation of China under Grant 61273246.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 281–290, 2013.

where $\boldsymbol{x}_i \in R^d$ and $L_i \in 2^L$ represent the ith instance and its relevant label subset. Additionally, the complement of L_i, i.e., $\bar{L}_i = L - L_i$, indicates the irrelevant label subset. In the original d-dimensional input space, q linear discriminant functions are defined as,

$$f_k(\boldsymbol{x}) = \boldsymbol{w}_k^T \boldsymbol{x} + b_k, k = 1, 2, ..., q. \tag{2}$$

It is desirable that any relevant label should be ranked higher than any irrelevant one. Through considering all possible pairwise constraints between relevant labels and irrelevant ones of each instance, the primary form of Rank-SVM is formulated as follows,

$$\begin{aligned} \min\ & \tfrac{1}{2}\sum_{k=1}^{q} \boldsymbol{w}_k^T \boldsymbol{w}_k + C\sum_{i=1}^{l} \tfrac{1}{|L_i|\left|\bar{L}_i\right|} \sum_{(m,n)\in(L_i\times\bar{L}_i)} \xi_{imn}, \\ \text{s.t.}\ & \left(\boldsymbol{w}_m - \boldsymbol{w}_n\right)^T \boldsymbol{x}_i + (b_m - b_n) \geq 1 - \xi_{imn}, \\ & \xi_{imn} \geq 0, (m,n) \in (L_i \times \bar{L}_i), i = 1, ..., l, \end{aligned} \tag{3}$$

where C is a positive regularization constant and ξ_{imn} represent the slack variables. The dual version of (3) becomes,

$$\begin{aligned} \min\ & F = \tfrac{1}{2}\sum_{k=1}^{q}\sum_{i,i'=1}^{l} \beta_{ki}\beta_{ki'}\left(\boldsymbol{x}_i^T \boldsymbol{x}_{i'}\right) - \sum_{i=1}^{l}\sum_{(m,n)\in(L_i\times\bar{L}_i)} \alpha_{imn}, \\ \text{s.t.}\ & \sum_{i=1}^{l}\sum_{(m,n)\in(L_i\times\bar{L}_i)} c_{imn}^k \alpha_{imn} = 0, k = 1, ..., q, \\ & 0 \leq \alpha_{imn} \leq C_i, (m,n) \in (L_i \times \bar{L}_i), i = 1, ..., l, \end{aligned} \tag{4}$$

with,

$$\begin{aligned} c_{imn}^k &= \begin{cases} 1, & \text{if } k = m, \\ -1, & \text{if } k = n, \\ 0, & \text{otherwise}, \end{cases} \\ \beta_{ki} &= \sum_{(m,n)\in(L_i\times\bar{L}_i)} c_{imn}^k \alpha_{imn}, \\ \boldsymbol{w}_k &= \sum_{i=1}^{l} \beta_{ki}\boldsymbol{x}_i, \\ C_i &= \tfrac{1}{|L_i|\left|\bar{L}_i\right|} C. \end{aligned} \tag{5}$$

We observe that the ith training instance is associated with a set of variables of the size $l_i = |L_i| \times \left|\bar{L}_i\right|$ to be solved, and the total number of variables is identical to $l_t = \sum_{i=1}^{l} l_i$. Therefore, Rank-SVM is a large-scale quadratic programming problem with q equality constraints and l_t box ones. Due to the complicated constraints, Rank-SVM is now solved by Frank-Wolfe method (FWM) [7, 8], in which the same scale linear programming problem is handled at each iteration.

The block coordinate descent method (BCDM) partitions all variables to be solved into some manageable blocks and updates a single block only at each iteration when the remaining blocks are fixed [9]. As we known, binary SVM is widely optimized by sequential minimization optimization (SMO), in which

a sub-problem with two variables are solved at each iteration [10, 11]. In [12], a coordinate gradient descent method (CGDM) was proposed for binary SVM. Essentially, such two methods are a greedy BCDM form since the gradient information is used to select two variables. Recently, a random BCDM (RBCDM) is presented for binary SVM [13], where two variables at least are chosen randomly at each iteration. It is also commended to apply RBCDM to some composite problems with many equality and box constraints, but no numerical example was provided [13]. In this paper, we utilize RBCDM to solve the above Rank-SVM, in which a smaller scale quadratic programming sub-problem is handled at each iteration. Our experiments on three data sets demonstrate that RBCDM is averagely 3 times faster than FWM for Rank-SVM, and Rank-SVM is a powerful candidate for multi-label classification, compared with the other four multi-label classifiers.

This paper is organized as follows. In Sections 2, FWM for Rank-SVM is reviewed. Our RBCDM for Rank-SVM is designed and implemented in Section 3. We experimentally analyze the convergence of RBCDM and FWM for Rank-SVM, and then compare Rank-SVM with the other four classifiers, in Section 4. Finally, this paper ends with some conclusions.

2 Frank-Wolfe Method for Rank-SVM

Let the entire solution vector $\boldsymbol{\alpha}$ and the gradient one $\boldsymbol{g}$ for (4) be,

$$\begin{aligned} \boldsymbol{\alpha} &= \left[\alpha_{imn} | (m,n) \in (L_i \times \bar{L}_i), i = 1, ..., l\right]^T, \\ \boldsymbol{g} &= \left[g_{imn} | (m,n) \in (L_i \times \bar{L}_i), i = 1, ..., l\right]^T, \end{aligned} \tag{6}$$

where,

$$\begin{aligned} g_{imn} &= \frac{\partial F}{\partial \alpha_{imn}} = \sum_{k=1}^{q} c_{imn}^k \sum_{i'=1}^{l} \beta_{ki'} (\mathbf{x}_i^T \mathbf{x}_{i'}) - 1 = \sum_{k=1}^{q} c_{imn}^k f_{ki} - 1, \\ f_{ki} &= \sum_{i'=1}^{l} \beta_{ki'} (\mathbf{x}_i^T \mathbf{x}_{i'}). \end{aligned} \tag{7}$$

The Frank-Wolfe method (FWM) is a simple and classical first order feasible direction optimization method [7, 8]. FWM generates a sequence of feasible vectors $\{\boldsymbol{\alpha}^{(t)}\}(t = 1, 2, ...)$ using a linear search $\boldsymbol{\alpha}^{(t+1)} = \boldsymbol{\alpha}^{(t)} + \lambda^{(t)} \boldsymbol{d}^{(t)}$, in which $\lambda^{(t)} \in [0, 1]$ is a step size and $\boldsymbol{d}^{(t)} = \bar{\boldsymbol{\alpha}}^{(t)} - \boldsymbol{\alpha}^{(t)}$ a feasible descent direction satisfying $z^{(t)} = (\boldsymbol{d}^{(t)})^T \boldsymbol{g}^{(t)} < 0$. To find out a best feasible direction, i.e., the best $\bar{\boldsymbol{\alpha}}^{(t)}$, FWM utilizes the first order Taylor series expansion of $F(\boldsymbol{\alpha})$ around the vector $\boldsymbol{\alpha}^{(t)}$ and then solves the following linear programming (LP) problem,

$$\begin{aligned} &\min \ \left(\bar{\boldsymbol{\alpha}}^{(t)}\right)^T \mathbf{g}^{(t)} = \sum_{i=1}^{l} \sum_{(m,n) \in (L_i \times \bar{L}_i)} \bar{\alpha}_{imn}^{(t)} g_{imn}^{(t)}, \\ &\text{s.t.} \ \sum_{i=1}^{l} \sum_{(m,n) \in (L_i \times \bar{L}_i)} c_{imn}^k \bar{\alpha}_{imn}^{(t)} = 0, k = 1, ..., q, \\ &\quad 0 \le \bar{\alpha}_{imn}^{(t)} \le C_i, (m,n) \in (L_i \times \bar{L}_i), i = 1, ..., l. \end{aligned} \tag{8}$$

Algorithm 1. FWM for Rank-SVM

1. Set two stopping criteria: M and ε.
2. Initialize $\alpha_{imn}^{(1)}$, $\beta_{ki}^{(1)}$, and $f_{ki}^{(1)}$ to be 0, and $g_{imn}^{(1)} = -1$.
3. For t=1,2,...,M.
   3-1. Solve the LP problem (8)
   3-2. Calculate $\boldsymbol{d}^{(t)}$ and $z^{(t)}$.
   3-3. (Option) if $|z^{(t)}| \leq \varepsilon$, then stop.
   3-4. Estimate the step size $\lambda^{(t)}$ using (9)
   3-5. Update $\boldsymbol{\alpha}^{(t+1)} = \boldsymbol{\alpha}^{(t)} + \lambda^{(t)}\boldsymbol{d}^{(t)}$.
   3-6. Re-calculate $\beta_{ki}^{(t+1)}$, $f_{ki}^{(t+1)}$ and $g_{imn}^{(t+1)}$.
   3-7. If the stopping condition (11) is satisfied, then stop.

This LP problem is usually optimized by simplex or interior point method. It has been proved that FWM has a sub-linear convergence rate [8].

According to (5) and $\bar{\boldsymbol{\alpha}}^{(t)}$, we calculate $\bar{\beta}_{ki}^{(t)}$ correspondingly. For some widely used Mercer kernels, e.g., linear, polynomial and RBF kernels, the objective function $F(\boldsymbol{\alpha})$ of Rank-SVM is strictly convex generally. In this case, the step size has a closed form, i.e.,

$$\lambda^{(t)} = \min\left\{1, -z^{(t)} \Big/ \sum_{k=1}^{q}\sum_{i,i'=1}^{l}\left(\bar{\beta}_{ki}^{(t)} - \beta_{ki}^{(t)}\right)\left(\bar{\beta}_{ki'}^{(t)} - \beta_{ki'}^{(t)}\right)(\mathbf{x}_i^T\mathbf{x}_{i'})\right\}. \quad (9)$$

Additionally, if $F(\boldsymbol{\alpha})$ has an optimal value $F(\boldsymbol{\alpha}^*)$, then

$$F(\boldsymbol{\alpha}^{(t)}) - F(\boldsymbol{\alpha}^*) \leq -z^{(t)}. \quad (10)$$

Therefore, usually $|z^{(t)}| \leq \varepsilon$ is used as a stopping condition in FWM. Note that this condition is related to the upper bounds C_i. Fairly, we define a relative stopping condition in this study,

$$\sqrt{\sum_{k=1}^{q}\sum_{i=1}^{l}\left(\beta_{ki}^{(t+1)} - \beta_{ki}^{(t)}\right)^2} \Bigg/ \sqrt{\sum_{k=1}^{q}\sum_{i=1}^{l}\left(\beta_{ki}^{(t+1)}\right)^2} \leq \varepsilon. \quad (11)$$

The FWM for Rank-SVM [6] is summarized in Algorithm 1, where M indicates the maximal number of iterations or epochs. In Algorithm 1, the most expensive step is to solve the LP problem, whose time complexity is $O(ql_t^2)$ for simplex method.

3 Random Block Coordinate Descent Method for Rank-SVM

The block coordinate descent method (BCDM) splits all variables to be solved into some manageable blocks, and then updates a single block only at each iteration when the remaining blocks are fixed [9]. In its random version (RBCDM),

Algorithm 2. RBCDM for Rank-SVM

1. Set two stopping criteria: M and ε.
2. Initialize α_{imn}, β_{ki}, and f_{ki} to be 0.
3. For $t = 1, 2, ..., M$.
   3-1. Permute all variables and then divide them into $M' = l_t/p$ blocks.
   3-2. For $t' = 1, 2, ..., M'$.
      3-2-1. Construct the QP sub-problem.
      3-2-2. Solve this QP problem using FWM listed in Algorithm 1.
      3-2-3. Update α_{imn}, β_{imn}, and f_{ki}.
   3-3. If the stopping condition (11) is satisfied, then stop.

the variables in a single block are selected randomly. In this section, we apply RBCDM to solve Rank-SVM.

Due to q equality constraints in Rank-SVM, the size of block is identical to q+1 at least. Assume that $p \geq q + 1$ variables are chosen randomly, and their indexes are denoted by $W = \{imn\}$, where $|W| = p$. Additionally, the indexes of the other variables are represented by N.

According to W and N, we permute the elements of α_{imn}, and then divide them into two parts,

$$\alpha_{imn} = \begin{cases} \alpha_{imn}^{W}, imn \in W, \\ \alpha_{imn}^{N}, imn \in N. \end{cases} \tag{12}$$

Now these variables in W are optimized and the others are fixed, i.e.,

$$\alpha_{imn}^{(t+1)} \Leftarrow \begin{cases} \alpha_{imn}^{W(t)} + \Delta\alpha_{imn}^{W(t)}, imn \in W, \\ \alpha_{imn}^{N(t)}, imn \in N. \end{cases} \tag{13}$$

Correspondingly, through isolating $c_{imn}^{kW(t)}$, $g_{imn}^{W(t)}$, $\beta_{ki}^{W(t)}$, and $C_i^{W(t)}$, we build a optimization subproblem of Rank-SVM as,

$$\begin{aligned} \min\ & \tfrac{1}{2} \sum_{k=1}^{q} \sum_{i,i' \in W} \Delta\beta_{ki}^{W(t)} \Delta\beta_{ki'}^{W(t)} \left(\boldsymbol{x}_i^T \boldsymbol{x}_{i'}\right) - \sum_{imn \in W} g_{imn}^{W(t)} \Delta\alpha_{imn}^{W(t)}, \\ \text{s.t.}\ & \sum_{imn \in W} c_{imn}^{kW(t)} \Delta\alpha_{imn}^{W(t)} = 0, k = 1, ..., q, \\ & 0 \leq \alpha_{imn}^{W(t)} + \Delta\alpha_{imn}^{W(t)} \leq C_i^{W(t)}, imn \in W, \end{aligned} \tag{14}$$

where $\Delta\beta_{ki}^{(t)}$ is associated with $\Delta\alpha_{imn}^{(t)}$ via (5). In this study, we use FWM in the above section to solve this sub-problem. With $\Delta\alpha_{imn}^{(t)}$ and W, we estimate $\Delta\beta_{ki}^{(t)}$, and then recursively update $\beta_{ki}^{(t+1)}$ and $f_{ki}^{(t+1)}$. Additionally, we calculate the gradients $g_{imn}^{(t+1)}$,$(imn \in W)$ using (7) temporarily rather than maintaining the entire gradient vector $\boldsymbol{g}$.

Our training procedure based on RBCDM for Rank-SVM is listed in Algorithm 2, in which, at each epoch, all l_t variables are permuted randomly and split

Table 1. Statistics of three benchmark data sets

Data set	Train	Test	Feature	Class	Average label	Variable	γ in RBF
Emotions	391	202	72	6	1.87	2793	2^{-1}
Scene	1211	1196	294	6	1.07	6278	2^{-3}
Yeast	1500	917	103	14	4.24	58248	2^{0}

Table 2. Comparison of RBCDM and FWM for Rank-SVM on three data sets

Method	FWM	RBCDM (the size of the active block = $z(q+1)$)					
		z=1	z=10	z=20	z=30	z=40	z=50
Emotion							
Fval	-105.88	-105.76	**-105.89**	**-105.89**	**-105.89**	**-105.89**	**-105.89**
Epoch	125	136	27	**15**	**15**	17	**15**
Time	8.1	5.1	5.3	4.2	**4.0**	4.7	4.7
F1	63.65	**64.19**	63.50	63.50	63.50	63.36	63.50
Scene							
Fval	-174.92	-174.86	**-174.96**	**-174.96**	**-174.96**	**-174.96**	**-174.96**
Epoch	227	180	23	**20**	23	22	23
Time	70.9	178.7	18.5	**17.1**	19.3	19.1	21.1
F1	**78.10**	**78.10**	77.92	77.82	77.82	**78.10**	77.82
Yeast							
Fval	**-399.53**	-397.07	-399.35	-399.45	-399.49	-399.50	-399.51
Epoch	61	374	71	37	38	**21**	23
Time	1433.0	3380.7	498.6	291.2	335.7	**212.4**	257.7
F1	59.03	59.04	59.05	**59.21**	59.03	59.11	59.20

into $M' = l_t/p$ blocks, and then each block are optimized by FWM sequentially. The convergence rate of our RBCDM for Rank-SVM could be proved by the theorem 4 in [13].

4 Experiments

In this section, we analyze the convergence of RBCDM and FWM for Rank-SVM, and then compare the performance of Rank-SVM with four existing multi-label classifiers: BP-MLL [14], ML-RBF [15], ML-NB [16] and ML-kNN [17], on three data sets (Emotions, Scene and Yeast) from [18]. Table 1 lists some basic statistics of three sets, including the numbers of training instances (Train), testing instances (Test), features (Feature) and classes (Class) respectively, average labels, the number of variables to be solved in Rank-SVM (Variable), and the optimal scale factor γ in RBF kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma||\boldsymbol{x} - \boldsymbol{y}||_2^2)$. Our RBCDM and FWM for Rank-SVM are coded using C++ language in MLC-SVM software package [19], where the free LPSOL5.5 from [20] is used as a LP solver. The free Matlab software of the other four classifiers is downloaded from [21].

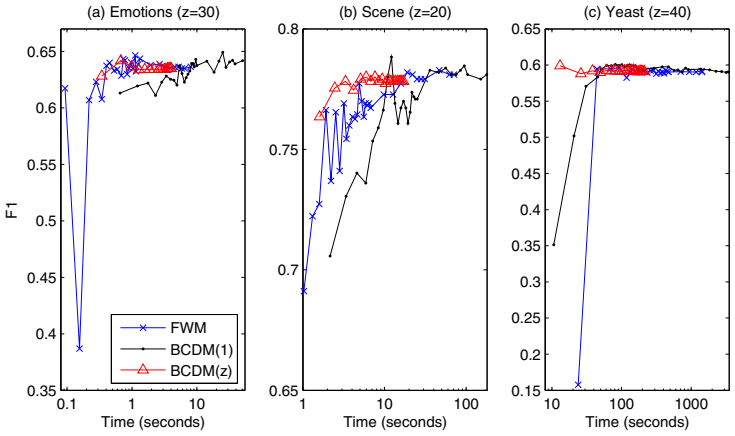


Fig. 1. The convergence of RBCDM and FWM for Rank-SVM on three data sets

4.1 Convergence Analysis of RBCDM and FWM for Rank-SVM

We divide each training set in Table 1 into two parts according to the ratio of 70% to 30%. The first one is used as a training subset to train a Rank-SVM model, and the other as a validating subset to evaluate the classification performance.

Given C=1.0, M=50 and $\epsilon = 10^{-3}$, which are the default setting in [5],[14–17], we execute RBCDM with z=1 for Rank-SVM using 13 γ's: $2^2, 2^1, ..., 2^{-9}, 2^{-10}$ and select the optimal γ with the highest F1 measure for each data set, as shown in the last column of Table 1.

With the optimal γ's in Table 1, we set $M = 1, 2, ..., 19, 20, 30, ..., 90, 100, 150, 200, 250, 350, 400$, and observe the optimization performance of FWM and RBCDM with the different size of block $p = z(q + 1), z = 1, 10, 20, 30, 40, 50$.

When $\epsilon = 10^{-3}$ is just satisfied, the objective function value (Fval), the number of epochs (Epoch), and training time (Time) in seconds from training subsets, and F1 value from validating subsets, are listed in Table 2, in which the best values are highlighted by boldface. It is found out that two optimization techniques can obtain a satisfactory performance, since the maximal relative difference of the objective function value and F1 are 0.62% on Yeast and 1.29% on Emotions, respectively. Except for z=1 (i.e., the size of block is q+1), our RBCDM needs less training time and less epochs than FWM. At the optimal cases, our RBCDM runs 1.03, 3.15 and 5.75 times faster than FWM on Emotions (z=30), Scene (30) and Yeast (40) respectively. Correspondingly, we recommend z=40 as a proper default setting.

To observe the convergence of RBCDM and FWM for Rank-SVM, we estimate the training time and F1 value of different epochs, as shown in Fig.1, where three curves from FWM and RBCDM with z=1 and the optimal z value are plotted, and the training procedure is terminated by $\epsilon = 10^{-3}$. FWM oscillates and then tend to be stable. RBCDM with z=1 converges the slowest among three methods.

Table 3. Performance comparison of six methods on three data sets

Method	RBCDM	FWM	BP-MLL	ML-RBF	ML-NB	ML-kNN
Emotion						
Hamming loss ↓	0.2038(2)	**0.2022**(1)	0.2170(5)	0.2079(3)	0.2294(6)	0.2088(4)
Accuracy ↑	0.5540(2)	**0.5573**(1)	0.5363(4)	0.5371(3)	0.4468(6)	0.5008(5)
F1 ↑	0.6343(2)	**0.6361**(1)	0.6234(3)	0.6104(4)	0.5262(6)	0.5847(5)
Precision ↑	0.6988(2)	**0.7005**(1)	0.6675(4)	0.6733(3)	0.6114(6)	0.6576(5)
Recall ↑	0.6320(3)	0.6337(2)	**0.6469**(1)	0.6056(4)	0.5124(6)	0.5734(5)
Subset accuracy ↑	0.3069(3)	**0.3168**(1)	0.2475(4)	0.3119(2)	0.2079(6)	0.2376(5)
Average rank ↓	2.33	**1.17**	3.50	3.17	6.00	4.83
Scene						
Hamming loss ↓	**0.0888**(1)	0.0900(2)	0.2907(6)	0.0945(3)	0.1242(5)	0.0989(4)
Accuracy ↑	**0.7398**(1)	0.7361(2)	0.1682(6)	0.5949(4)	0.5325(5)	0.6293(3)
F1 ↑	**0.7655**(1)	0.7616(2)	0.1704(6)	0.6084(4)	0.5668(5)	0.6483(3)
Precision ↑	**0.7747**(1)	0.7713(2)	0.1732(6)	0.6221(4)	0.5552(5)	0.6605(3)
Recall ↑	**0.7818**(1)	0.7772(2)	0.1706(6)	0.6083(5)	0.6133(4)	0.6547(3)
Subset accuracy ↑	**0.6630**(1)	0.6597(2)	0.1622(6)	0.5544(4)	0.4323(5)	0.5727(3)
Average rank ↓	**1.00**	2.00	6.00	4.00	4.83	3.17
Yeast						
Hamming loss ↓	0.1988(2)	0.1990(3)	0.2086(6)	0.2004(4)	0.2066(5)	**0.1980**(1)
Accuracy ↑	0.5092(2.5)	0.5092(2.5)	**0.5185**(1)	0.5038(4)	0.4890(6)	0.4920(5)
F1 ↑	0.6192(2)	0.6189(3)	**0.6313**(1)	0.6081(4)	0.5995(5)	0.5993(6)
Precision ↑	0.7094(2)	0.7089(3)	0.6680(6)	0.7024(4)	0.6982(5)	**0.7322**(1)
Recall ↑	0.5949(2)	0.5944(3)	**0.6443**(1)	0.5832(4)	0.5698(5)	0.5491(6)
Subset accuracy ↑	0.1658(3)	0.1679(2)	0.1527(5.5)	**0.1778**(1)	0.1527(5.5)	0.1592(4)
Average rank ↓	**2.25**	2.75	3.42	3.50	5.25	3.83

Attractively, RBCDM with the optimal z value can quickly achieve to be stable, which implies that RBCDM has a good convergence rate.

4.2 Performance Comparison

In this sub-section, we evaluate the performance of Rank-SVM with RBCDM and FWM, BP-MLL [14], ML-RBF [15], ML-NB [16] and ML-kNN [17] using a train-test mode. Additionally, for the last four methods, we accept their recommended parameter settings. For BP-MLL, the number of hidden neurons is set to be 20% of the number of input neurons, the learning rate is 0.05, the number of training epochs is fixed to be 100, and the regularization constant is 0.1. For ML-kNN, the smooth factor $s = 1$ and nearest instances $k = 10$ with Euclidean distance. For ML-RBF, the fraction factor $\alpha = 0.01$ and the scaling factor $\mu = 1.0$. For ML-NB, the fraction of remaining features after PCA is set to be 0.3.

We retrain six classifiers on three training sets and then estimate their performance on three testing sets in Table 1. In this study, six indicative instance-based measures [3, 4] are used, i.e., Hamming loss, accuracy, F1, precision, recall and

subset accuracy. Our experimental results are listed in Table 3, where ↑ means that the higher a measure is, the better a method performs, and reversely for ↓.

To compare these classifiers fairly and comprehensively, we rank six methods using 1-6 in the brackets for each measure, and calculate the average rank of each method over six measures [22], as listed in Table 3. On Emotions, FWM for Rank-SVM performs the best on five measures and achieves the best rank, but RBCDM for Rank-SVM obtains the second best rank. About Scene, our RBCDM for Rank-SVM works the best on all six measures. As per Yeast, BP-MLL performs the best and worst on three measures respectively, which implies that BP-MLL is not stable. Further, our RBCDM for Rank-SVM achieves the best rank on Yeast.

According to the average rank over three data sets, we can sort six techniques in descending order as RBCDM for Rank-SVM (**1.86**), FWM for Rank-SVM (1.97), ML-RBF(3.56), BP-MLL(4.31), ML-kNN(3.94) and ML-NB(5.36). It can be concluded that Rank-SVM is a powerful candidate for multi-label classification.

5 Conclusions

In this paper, we propose a random block coordinate descent method for multi-label support vector machine to speed up its training procedure. The convergence analysis on three data sets illustrate that our method runs averagely 3.31 times faster that Frank-Wolfe method. Further, the performance of six instance-based measures demonstrates that Rank-SVM is still a powerful candidate for multi-label classification, compared with the other four existing methods. In future, we will conduct more detailed experiments on more data sets to validate our novel training technique.

References

1. Boutell, M.R., Lu, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. Pattern Recognition 37(9), 1757–1771 (2004)
2. Tsoumakas, G., Katakis, I.: Multi-label classification: an overview. International Journal of Data Warehousing and Mining 3(3), 1–13 (2007)
3. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, 2nd edn., pp. 667–685. Springer, Heidelberg (2010)
4. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE Transactions on Knowledge and Data Engineering (in press, 2013), doi:10.1109/TKDE.2013.39
5. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: 14th Annual Conference on Neural Information Processing Systems, pp. 681–687. MIT Press, Cambridge (2002)
6. Elisseeff, A., Weston, J.: Kernel methods for multi-labelled classification and categorical regression problems. Technical Report, BIOwulf Technologies (2001), `http://www.kyb.tuebingen.mpg.de/bs/people/weston/publications`
7. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Research Logistic Quarterly 3(1-2), 95–110 (1956)

8. Guelat, J., Marcotte, P.: Some comments on Wolfe's away step. Mathematical Programming 34(1), 110–119 (1986)
9. Wen, Z.W., Goldfarb, D., Scheinberg, K.: Block coordinate descent methods for seimdefinate programming. In: Anjos, M.F., Lasserre, J.B. (eds.) Handbook on Semidefinite, Conic and Polynomial Optimization, pp. 533–564. Springer, Heidelberg (2012)
10. Platt, J.: Sequential minimal optimization: a fast algorithm for training support vector machines. In: Scholkopf, B., Burges, C.J.C., Smola, A.J. (eds.) Advances in Kernel Methods - Support Vector Learning, pp. 185–208. MIT Press, Cambridge (1999)
11. Fan, R.E., Chen, P.H., Lin, C.J.: Working set selection using second order information for training support vector machines. Journal of Machine Learning Research 6, 1889–1918 (2005)
12. Tseng, P., Yun, S.: A coordinate gradient descent method for linearly constrained smooth optimization and support vetcor machines training. Computational Optimization and Applications 47(2), 179–206 (2010)
13. Necoara, I., Patrascu, A.: A random coordinate descent algorithm for optimization problems with composite objective function and linear coupled constraints. Technical Report, Department of Automatic Control and Systems Engineering, University Politehnica Bucharest, Romania (2013), `http://arXiv.1302.3074v1`
14. Zhang, M.L., Zhou, Z.H.: Multilabel neural networks with application to function genomics and text categorization. IEEE Transactions on Knowledge and Data Engineering 18(10), 1338–1351 (2006)
15. Zhang, M.L.: ML-RBF: RBF neural networks for multi-label learning. Neural Processing Letters 29(2), 61–74 (2009)
16. Zhang, M.L., Pena, J.M., Robles, V.: Feature selection for multi-label naive Bayes classification. Information Sciences 179(19), 3218–3229 (2009)
17. Zhang, M.L., Zhou, Z.H.: ML-kNN: a lazy learning approach to multi-label learning. Pattern Recognition 40(5), 2038–2048 (2007)
18. Multi-label data sets, `http://mulan.sourceforge.net/datasets.html`
19. Multi-label suport vector machine C/C++ software of MLC-SVM, `http://computer.njnu.edu.cn/Lab/LABIC/LABIC_software.html`
20. Linear programming solver: LPSOL5.5, `http://lpsolve.soureforge.net`
21. Multi-label Matlab software of BP-MLL, ML-RBF, ML-NB and ML-kNN, `http://cse.seu.edu.cn/people/zhangml`
22. Brazdil, P.B., Soares, C.: A comparison of ranking methods for classification algorithm selection. In: Lopez de Mantaras, R., Plaza, E. (eds.) ECML 2000. LNCS (LNAI), vol. 1810, pp. 63–75. Springer, Heidelberg (2000)

Generalization Ability of Chaotic Complex-Valued Multidirectional Associative Memory with Adaptive Scaling Factor

Takuya Chino and Yuko Osana

Tokyo University of Technology,
1401-1 Katakura, Hachioji, Tokyo, Japan
osana@stf.teu.ac.jp

Abstract. In this paper, we propose a Chaotic Complex-valued Multidirectional Associative Memory (CCMAM) with adaptive scaling factor and investigate its generalization ability for network size. The proposed model is based on the conventional CCMAM with variable scaling factor and can realize one-to-many associations of M-tuple multi-valued patterns. In the proposed model, the scaling factor of refractoriness is determined based on not only the time but also the internal states of neurons. We carried out a series of computer experiments and confirmed that the proposed model can determine the scaling factor of refractoriness automatically in various size networks.

1 Introduction

As the model which can deal with multi-valued patterns, the complex-valued neural network[1] has been proposed. Moreover, we modified the complex-valued neural network by introducing chaotic complex-valued neurons[2] and proposed some models which can realize one-to-many associations of multi-valued patterns [3]–[8]. In these models, the association of multi-valued patterns is realized by complex-valued neurons, and one-to-many association is realized by chaotic complex-valued neurons. Moreover, in the Chaotic Complex-valued Multidirectional Associative Memory (CCMAM) with variable scaling factor[6][7], one-to-many association ability is improved by introducing variable scaling factor. However, in these models, their property is very sensitive to chaotic complex-valued neuron parameters, and in most cases, these parameters are determined based on the designer's experiments or trial and errors.

In this paper, we propose the Chaotic Complex-valued Multidirectional Associative Memory (CCMAM) with adaptive scaling factor. In the proposed model, the appropriate parameter (scaling factor of refractoriness) can be determined based on the internal states of neurons automatically.

2 Chaotic Complex-Valued Multidirectional Associative Memory with Adaptive Scaling Factor

Here, I explain the proposed Chaotic Complex-Valued Multidirectional Associative Memory (CCMAM) with adaptive scaling factor. The proposed model

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 291–298, 2013.

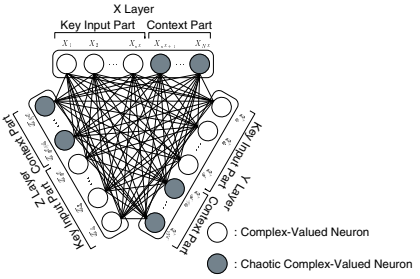


Fig. 1. Structure of Proposed CCMAM with Adaptive Scaling Factor

is based on the conventional Chaotic Complex-Valued Multidirectional Associative Memory with variable scaling factor[6][7], and can realize one-to-many association of M-tuple multi-valued patterns.

2.1 Structure

The proposed model has three or more layers as similar as the conventional CCMAM with variable scaling factor. Figure 1 shows the structure of the proposed model which has three layers (X Layer, Y Layer and Z Layer). Each layer consists of two parts; (1) key input part composed of complex-valued neuron models[1] and (2) context part composed of chaotic complex-valued neuron models[2]. In this model, since the chaotic complex-valued neuron models in the context part change their states by chaos, one-to-many association can be realized.

2.2 Learning Process

In the proposed model, pattern sets are memorized by the orthogonal learning. In the proposed model which has M layers, the connection weights from the layer x to the layer y is given by

$$\boldsymbol{w}^{yx} = \boldsymbol{X}_x(\boldsymbol{X}_y^* \boldsymbol{X}_y)^{-1} \boldsymbol{X}_y^* \tag{1}$$

where * shows the conjugate transpose, -1 shows the inverse, and $\boldsymbol{X}_x$ and $\boldsymbol{X}_y$ are the training pattern matrixes which are memorized in the layer x and the layer y, and are given by

$$\boldsymbol{X}_x = \{\boldsymbol{X}_x^{(1)}, \cdots, \boldsymbol{X}_x^{(p)}, \cdots, \boldsymbol{X}_x^{(P)}\} \tag{2}$$

$$\boldsymbol{X}_y = \{\boldsymbol{X}_y^{(1)}, \cdots, \boldsymbol{X}_y^{(p)}, \cdots, \boldsymbol{X}_y^{(P)}\} \tag{3}$$

where $\boldsymbol{X}_x^{(p)}$ is the pattern p which is stored in the layer x, $\boldsymbol{X}_y^{(p)}$ is the pattern p which is stored in the layer y and P is the number of the training pattern sets.

In the orthogonal learning, since the stored common pattern cause superimposed pattern in the recall process, the pattern sets including one-to-many

relation can not be memorized. In the proposed model, each learning pattern is memorized together with its own contextual information in order to memorize the training set including one-to-many relations as similar as the conventional CCMAM with variable scaling factor. Here, the contextual information patterns are generated randomly.

2.3 Recall Process

In the recall process, only neurons in the key input part receives input in the first step. This is because we assume that contextual information is usually unknown for users. In the proposed model, since the chaotic complex-valued neurons in the context part change their states by chaos, plural patterns corresponding to the input common pattern can be recalled.

Step 1 : Input to Layer x

The input pattern is given to the key input part in the layer x.

Step 2 : Propagation from Layer x **to Other Layers**

The information in the layer x is propagated to the key input part in other layers. The output of the neuron k in the key input part of the layer y ($y \neq x$) at the time t, $x_k^y(t)$ is calculated by

$$x_k^y(t) = f\left(\sum_{j=1}^{N^x} w_{kj}^{yx} x_j^x(t)\right) \tag{4}$$

where N^x is the number of neurons in the layer x, w_{kj}^{yx} is the connection weight from the neuron j in the layer x to the neuron k in the layer y, and $x_j^x(t)$ is the output of the neuron j in the layer x at the time t. $f(\cdot)$ is the output function which is given by

$$f(u) = \frac{\eta u}{\eta - 1.0 + |u|} \qquad (\eta \in \mathbb{R}) \tag{5}$$

where η is the constant ($\eta > 1$).

Step 3 : Propagation from Other Layers to Layer x

The information in other layers is propagated to the layer x. The output of the neuron j in the key input part of the layer x, $x_j^x(t+1)$, is given by

$$x_j^x(t+1) = f\left(\sum_{y \neq x}^{M}\left(\sum_{k=1}^{n^y} w_{jk}^{xy} x_k^y(t)\right) + vA_j^x\right) \tag{6}$$

where M is the number of layers, n^y is the number of neurons in the key input part of the layer y, w_{jk}^{xy} is the connection weight from the neuron k in the layer y

to the neuron j in the layer x, and v is the connection weight from the external input. A_j^x is the external input to the neuron j in the layer x and is given by

$$A_j^x = \begin{cases} 0, & t < t_{in} \\ \hat{x}_j^x(t_{in}), & t_{in} \leq t \end{cases} \tag{7}$$

$$t_{in} = \min \left\{ t \left| \sum_{j=1}^{n^x} (\hat{x}_j^x(t) - \hat{x}_j^x(t-1)) = 0 \right. \right\} \tag{8}$$

$$\hat{x}_j^x(t) = \underset{s}{\text{argmin}} (\omega^s - x_j^x(t))^* (\omega^s - x_j^x(t)) \quad (s = 0, 1, ..., S-1) \tag{9}$$

$$\omega = \exp\left(i \frac{2\pi}{S}\right) \tag{10}$$

where $\hat{x}_j^x(t)$ is the quantized output of the neuron j in the layer x at the time t, S is the number of states and i is the imaginary unit.

The output of the neuron j of the context part in the layer x, $x_j^x(t+1)$ is given by

$$x_j^x(t+1) = f\left(\sum_{y \neq x}^{M} \left(\sum_{k=1}^{n^y} w_{jk}^{xy} \sum_{d=0}^{t} k_m^d x_k^d(t-d) \right) - \alpha(t, I) \sum_{d=0}^{t} k_r^d x_j^x(t-d) \right) \tag{11}$$

where k_m and k_r are damping factors. And, $\alpha(t, I)$ is the scaling factor of refractoriness at the time t and the average internal state without refractoriness I. The average internal state without refractoriness I is given by

$$I = \frac{1}{N^x - n^x} \sum_{j=n^x+1}^{N^x} \frac{|\text{Re } u_j| + |\text{Im } u_j|}{2} \tag{12}$$

$$u_j = \sum_{y \neq x}^{M} \left(\sum_{k=1}^{n^y} w_{jk}^{xy} \sum_{d=0}^{t} k_m^d x_k^y(t-d) \right). \tag{13}$$

In Eq.(11), $\alpha(t, I)$ is given by

$$\alpha(t, I) = a(I) + b(a(I), S) \sin\left(c \cdot \frac{\pi}{12} \cdot t\right) \tag{14}$$

$$a(I) = 0.75 + 0.009(I - 1.0)^2 \tag{15}$$

$$b(a(I), S) = \begin{cases} a(I), & S = 4,\ 8 \\ a(I)/2, & S = 16. \end{cases} \tag{16}$$

Eqs.(15) and (16) are determined based on the experiment shown in 2.4.

Step 4 : Repeat

Steps 2 and **3** are repeated.

Table 1. Parameter Combination of a and b when Recall Rate is Highest

M	S	a	b
3	4	1.0	1.0
	8	1.0	0.5
	16	1.0	0.5
4	4	1.5	1.5
	8	1.5	1.0
	16	1.5	1.0
5	4	2.0	2.0
	8	2.0	1.0
	16	2.0	1.0

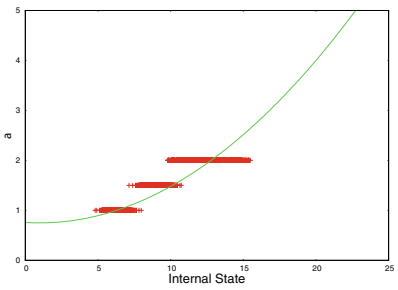


Fig. 2. Relation between Internal States and Parameter a when Recall Rate is High

2.4 Relation between One-to-Many Association Ability and Internal States

In the chaotic complex-valued neuron model, whether chaos occurs or not is determined based on the external input. In the CCMAM with variable scaling factor[6][7], the internal state of each neuron is calculated based on the input from other layers and the refractoriness of the neuron. So the input from other layers can be considered as the external input to the neuron.

In the conventional CCMAM with variable scaling factor, the scaling factor of refractoriness at the time t, $\alpha(t)$ is given by

$$\alpha(t) = a + b \cdot \sin\left(c \cdot \frac{\pi}{12} t\right) \tag{17}$$

and it is known that the combination of the parameters a and b affects one-to-many association ability[7][8].

Then, we examined the relation between one-to-many association ability and internal states in the conventional CCMAM with variable scaling factor which has 100 neurons in the key input part and 50 neurons in the context part in order to decide the method how to determine the scaling factor of refractoriness. We investigated the recall rate in various combination of a and b in $3\sim 5$ layered CCMAM with variable scaling factor.

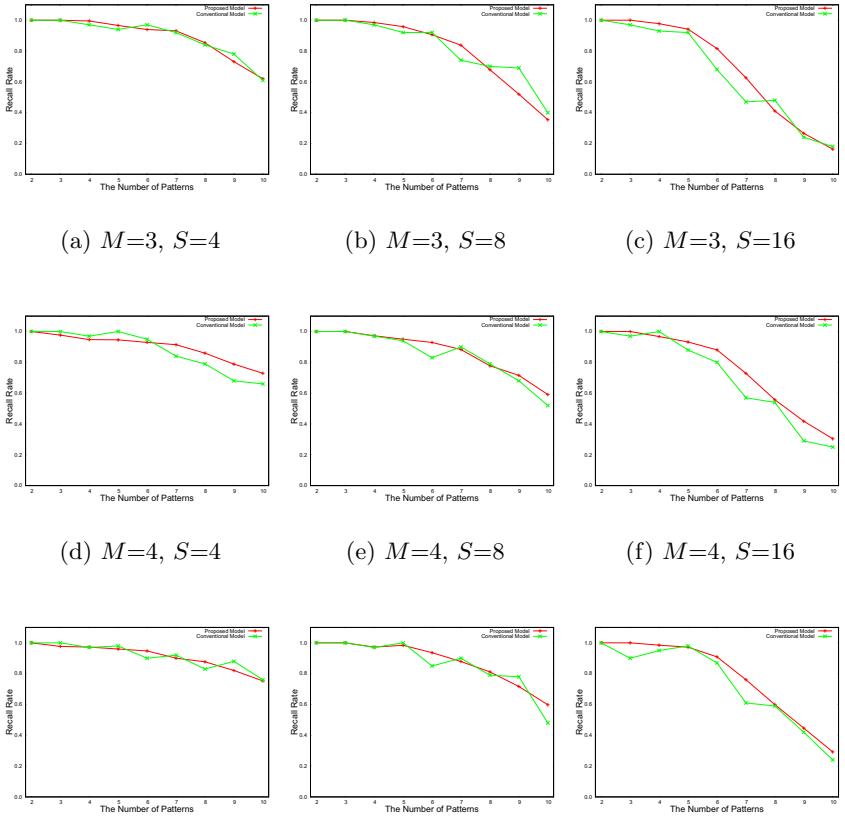
Table 1 shows the combination of a and b when the recall rate is highest in each condition. Figure 2 shows the relation between internal state without refractoriness and the appropriate parameter a. And the green line shows the function which determines the parameter a in the proposed model (Eq.(15)). This function is determined based on these experiments.

3 Computer Experiment Results

Here, we show the computer experiment results in order to demonstrate of effectiveness of the proposed model.

Here, we compared the one-to-many association ability in the 3∼5-layered proposed model with the well-turned 3∼5-layered conventional CCMAM with variable scaling factor[6][7]. Figure 3 shows the one-to-many association ability of the proposed model and the conventional model. As shown in this figure, the one-to-many association ability of the proposed model almost equals to that of the conventional model. Here, N_K is the number of neurons in the key input part, and N_C is the number of neurons in the context part.

Figure 4 shows the one-to-many association ability of the various size proposed model. As shown in this figure, the proposed model has good one-to-many association ability as similar as in the result shown in Fig. 3.

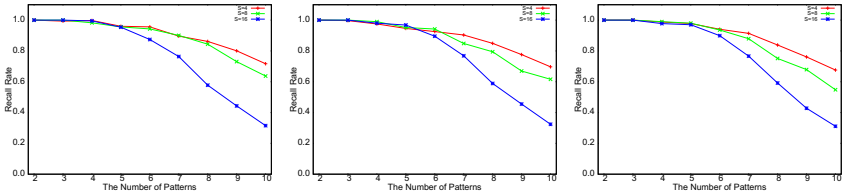


(a) M=3, S=4 (b) M=3, S=8 (c) M=3, S=16

(d) M=4, S=4 (e) M=4, S=8 (f) M=4, S=16

(g) M=5, S=4 (h) M=5, S=8 (i) M=5, S=16

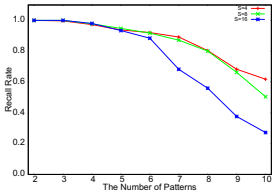
Fig. 3. One-to-Many Association Ability Comparison with Conventional Model (N_K = 100, N_C = 50)



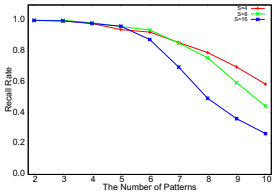
(a) M=3, N_K=200, N_C=50

(b) M=4, N_K=200, N_C=50

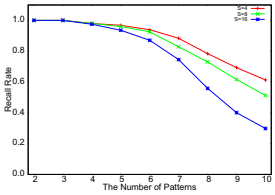
(c) M=5, N_K=200, N_C=50



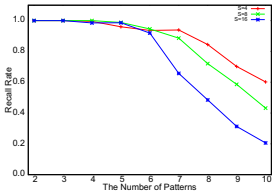
(d) M=3, N_K=300, N_C=50



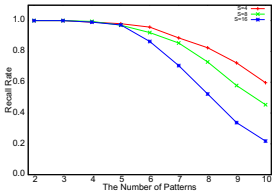
(e) M=4, N_K=300, N_C=50



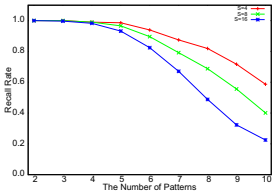
(f) M=5, N_K=300, N_C=50



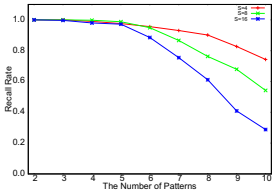
(g) M=3, N_K=100, N_C=20



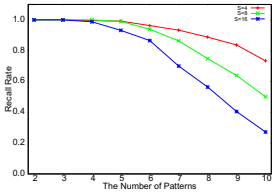
(h) M=4, N_K=100, N_C=20



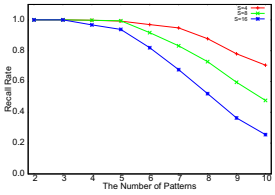
(i) M=5, N_K=100, N_C=20



(j) M=6, N_K=100, N_C=50



(k) M=7, N_K=100, N_C=50



(l) M=8, N_K=100, N_C=50

Fig. 4. One-to-Many Association Ability in Various Conditions

4 Conclusion

In this paper, we have proposed the Chaotic Complex-valued Multidirectional Associative Memory (CCMAM) with adaptive scaling factor and investigated its generalization ability for network size. The proposed model is based on the conventional CCMAM with variable scaling factor[6][7] and can realize one-to-many associations of M-tuple multi-valued patterns. In the proposed model, the scaling factor of refractoriness is determined based on not only the time but also the internal states of neurons. We carried out a series of computer experiments and confirmed that the proposed model can determine the scaling factor of refractoriness automatically in various size networks.

References

1. Jankowski, S., Lozowski, A., Zurada, J.M.: Complex-valued multistate neural associative memory. IEEE Trans. Neural Networks 7(6), 1491–1496 (1996)
2. Nakada, M., Osana, Y.: Chaotic complex-valued associative memory. In: Proceedings of International Symposium on Nonlinear Theory and its Applications, Vancouver (2007)
3. Yano, Y., Osana, Y.: Chaotic complex-valued bidirectional associative memory. In: Proceedings of IEEE and INNS International Joint Conference on Neural Networks, Atlanta (2009)
4. Yano, Y., Osana, Y.: Chaotic complex-valued bidirectional associative memory – one-to-many association ability –. In: Proceedings of International Symposium on Nonlinear Theory and its Applications, Sapporo (2009)
5. Shimizu, Y., Osana, Y.: Chaotic complex-valued multidirectional associative memory. In: Proceedings of IASTED Artificial Intelligence and Applications, Innsbruck (2010)
6. Yoshida, A., Osana, Y.: Chaotic complex-valued multidirectional associative memory with variable scaling factor. In: Honkela, T. (ed.) ICANN 2011, Part I. LNCS, vol. 6791, pp. 266–274. Springer, Heidelberg (2011)
7. Yoshida, A., Osana, Y.: Chaotic complex-valued multidirectional associative memory with variable scaling factor – One-to-many association ability –. In: Proceedings of IEEE and INNS International Joint Conference on Neural Networks (2011)
8. Onagi, M., Osana, Y.: Pattern dependency of one-to-many association ability in chaotic complex-valued multidirectional associative memory with variable scaling factor. In: Proceedings of IEEE and INNS International Joint Conference on Neural Networks, Brisbane (2012)
9. Aihara, K., Takabe, T., Toyoda, M.: Chaotic neural networks. Physics Letter A 144(6,7), 333–340 (1990)

Improved Multi-class Support Vector Machines Using Novel Methods of Model Selection and Feature Extraction

Takuya Kitamura and Kengo Ota

Toyama National College of Technology
13, Hongo-cho, Toyama, Japan
kitamura@nc-toyama.ac.jp

Abstract. In this paper, to improve the generalization capability of multi-class SVMs, we propose (1) a novel model selection and (2) feature extraction by SVMs. In (1), unlike the conventional model selection in multi-class SVMs, we determine hyper-parameters, which are kernel parameter and margin parameter, for each separating hyper-plane, separately. Namely, for each separating hyper-plane, we estimate the generalization capability and select optimal values of the hyper-parameters, separately. In (2), we define the weighted vectors of decision functions determined by training multi-class SVMs as the basis vector of the subspace, and we determine the separating hyper-planes in the subspace. Thus, we can determine the new separating hyper-planes during considering the all separating hyper-planes. Using multi-class benchmark data sets, we evaluate the effectiveness of the proposed methods over the conventional method.

Keywords: classification, model selection, sparse support vector machines.

1 Introduction

Support vector machines (SVMs) [1], [2], [3] perform very well for pattern classification problem [4] and the solution is sparse. The separating hyper-plane is determined in training SVMs so that margins of the separating hyper-planes are maximized. And, to enhance linear separability for nonlinear problem, input space is mapped into high dimensional space called feature space with kernel methods [3]. Then, it is necessary to select two optimal hyper-parameters which are a kernel parameter and a margin parameter. To select the hyper-parameters is called model selection. k-fold cross validation [3] is one of the major methods of the model selection. Since the standard SVMs are two-class classifier, it is necessary for multi-class problem to convert it into some two-class problems (e.g., one-against-all and pairwise formulation [3] etc.). Namely, there are multiple separating hyper-planes for multi-class problem. For all separating hyper-planes, each hyper-parameter is same value. However, these may not be optimal values for each separating hyper-plane. And, in training multi-class SVMs, we classify

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 299–306, 2013.

an input datum into a class by decision functions, but each decision function is separately-determined for each separating hyper-plane.

In this paper, for multi-class problem, we propose two methods, which are a novel model selection and a feature extraction using SVMs. In first method, for each two-class problem, we select hyper-parameters which are optimal values from the standpoint of the generalization capability, separately. Namely, with the training data, we estimate the generalization capability for each two-class problem instead of estimating that for multi-class problem, separately. Then, the complexity of model selection may not be increased in this method as compared to the conventional method because we solve the same optimization problem as that using the conventional method. In second method, at first, we obtain the weighted vector of the decision function for each two-class problem, like the conventional multi-class SVMs. Next, we define these as the basis vectors of the subspace. Namely, SVMs are used as the feature extraction. And we train SVMs in the subspace. Thus, we can determine separating hyper-plane, which can consider all separating hyper-plane in feature space, in the subspace.

This paper is organized as follows. In Section 2, we describe the conventional multi-class SVMs. In Section 3, we discuss the proposed methods. In Section 4, we demonstrate the effectiveness of the proposed methods through computer experiments using benchmark data sets. And we conclude our work in Section 5.

2 Multi-class Support Vector Machines

In this section, we will describe training multi-class SVMs and model selection in one-against-all formulation for multi-class problem.

2.1 Multi-class SVMs in One-against-All Formulation

SVMs are two-class classifier. So, for multi-class problem, we need to convert that problem into some two-class problems. One-against-all SVMs is one of the most major methods for multi-class SVMs. In one-against-all SVMs, an n-class problem is converted into n two-class problems. For each two-class problem, we determine a separating hyper-plane so that a class is separated from the others.

Let the number of classes, that of training data, the m-dimensional training vectors, and the kernel be n, M, $\boldsymbol{x}_j$ $(j = 1, ..., M)$, and $H(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{g}^{\mathrm{T}}(\boldsymbol{x})\boldsymbol{g}(\boldsymbol{x}')$ where $\boldsymbol{g}(\boldsymbol{x})$ is the mapping function into the l-dimensional feature space. For two-class problem that separate class i from the remaining classes, the decision function $D_i(\boldsymbol{x})$ is

$$D_i(\boldsymbol{x}) = \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{x}) + b_i, \tag{1}$$

where $\boldsymbol{w}_i$ and b_i are the l-dimensional weighted vector and the bias term. And, we formulate SVMs (L2-SVMs) for this problem as follows:

$$\begin{aligned} \min \quad & \tfrac{1}{2}\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i + \tfrac{C}{2}\textstyle\sum_{j=1}^{M}\xi_{ij}^2 && (2)\\ \text{s.t.} \quad & y_{ij}(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{x}_j) + b) \geq 1 - \xi_{ij} \quad \text{for} \quad j = 1, \ldots, M, && (3) \end{aligned}$$

where C and ξ_{ij} are the margin parameter which determines the trade-off between maximizing margins and minimizing misclassifications and the slack variable for $\boldsymbol{x}_j$, respectively. If $\boldsymbol{x}_j$ belongs to class i, $y_{ij} = 1$, otherwise $y = -1$. All the decision functions $D_i(\boldsymbol{x})$ $(i = 1, \ldots, n)$ are given by solving the optimization problem for n two-class problems. Then, an input datum $\boldsymbol{x}$ is classified into the class

$$\arg\max_i D_i(\boldsymbol{x}). \tag{4}$$

However, the decision function for each two-class problem is determined without that for the other two-class problems. So, we consider that to classify an input datum into a class with (4) is not good.

2.2 Model Selection

In training SVMs, we must select a margin parameter C, a type of kernels, and a kernel parameter. The parameters and this selection are called hyper-parameters and model selection. The model selection is done by estimating the generalization capability for each combination of available choices of the hyper-parameters with training dataset, t. In the following computer experiments, we use k-fold cross validation. k-fold cross validation is widely used method of the model selection.

In k-fold cross validation, at first, we divide randomly training dataset into k subsets. Here, the sizes of these are approximately equal. Next, we select a combination of the hyper-parameters. For this combination, we train SVMs using $k - 1$ subsets as training dataset and test using the remaining subset as test dataset. These procedures is repeated k times. The generalization capability of SVMs for the combination of the hyper-parameters is estimated by the average recognition rate of the test datasets. And, for each combination of the hyper-parameters, we repeat the procedures and estimate the generalization capabilities. Then, we select the classifier that realizes the highest generalization capability.

However, for multi-class problem, there are multiple separating hyper-planes and the selected hyper-parameters may be not optimal for each separating hyper-plane.

3 Improved Multi-class Support Vector Machines

In this section, to overcome the above problems, we will describe two proposed methods that are the model selection for each separating hyper-plane and the feature extraction using SVMs.

3.1 Model Selection for Each Separating Hyper-plane

In the proposed method, like the conventional k-fold cross validation, the training dataset is divided into k subset and we select a combination of the hyper-parameters. Using $k - 1$ subsets as the training datasets, we determine the decision function for each two-class problem. Unlike the conventional model selection,

the generalization capability for each separating hyper-plane determined by the decision function associated with it is estimated separately. Then, to estimate this, we obtain the recognition rate for each separating hyper-plane. In one-against-all formulation, i-th separating hyper-plane separates the data belonging to class i from that belonging to the other classes. For i-th separating hyper-plane, let the number of the test data in k-fold cross validation and that of test data that satisfy $y(\boldsymbol{x})D_i(\boldsymbol{x}) > 0$ be M_i and A_i. Here, if $\boldsymbol{x}$ belongs to class i, $y(\boldsymbol{x}) = 1$, otherwise $y(\boldsymbol{x}) = -1$. Then, the recognition rate R_i for i-th separating hyper-plane is defined as follows:

$$R_i = \frac{A_i}{M_i} \qquad \text{for i} = 1, \ldots, \text{n}. \tag{5}$$

These procedures are repeated k times and the average recognition rate R'_i $(i = 1, \ldots, n)$ is determined. For each combination of the hyper-parameters, the average recognition rate is determined by repeating these procedures. And we select the classifier that realizes the highest generalization capability for each separating hyper-plane, separately. In the following, we show the algorithm of the proposed model selection using p combination of hyper-parameters.

Step 1 Divide randomly the training dataset into approximately equal sized k subsets.

Step 2 Set $c = 1$, $d = 1$, and $i = 1$.

Step 3 Set the c-th subset as test dataset and the other $k - 1$ subsets as training dataset.

Step 4 Using the d-th combination of the hyper-parameters that are the margin parameter C and the kernel parameter, solve the optimization problem (2),(3) in the dual form. And calculate $D_i(\boldsymbol{x})$ by (1) for test dataset.

Step 5 Calculate the number of test data that satisfy $y(\boldsymbol{x})D_i(\boldsymbol{x}) > 0$ and set it to A_i.

Step 6 Using A_i determined in Step 5 and M_i, calculate R_{id} by (5).

Step 7 If $i \neq n$, set $i = i + 1$ and go to Step 4. If $i = n$, go to Step 8.

Step 8 If $c \neq k$, set $i = 1$ and $c = c + 1$, and go to Step 3. If $c = k$, calculate the average recognition R'_{id} by R_{id} $(i = 1, \ldots, n)$ determined by Step 7, and go to Step 9.

Step 9 If $d \neq p$, set $i, c = 1$ and $d = d + 1$, and go to Step 3. If $d = p$, set $i = 1$ and go to Step 10.

Step 10 Calculate q_i as follows:

$$q_i = \arg\max_d R'_{id}. \tag{6}$$

Step 11 Select the q_i-th combination of the hyper-parameters for i-th separating hyper-plane.

Step 12 If $i \neq n$, set $i = i + 1$ and go to Step 10. If $i = n$, terminate the algorithm.

3.2 Feature Extraction by SVMs in One-against-All Formulation

To overcome the problem of section 2.1, multi-class SVMs is used as the feature extraction. Namely, we let the weighted vectors, that obtained by training multi-class SVMs in one-against all formulation, be basis vectors of the n-dimensional subspace. And multi-class SVMs are retrained in this subspace. Then, the projection function $\boldsymbol{f}(\boldsymbol{x})$ of a input datum $\boldsymbol{x}$ into the subspace is determine as follows:

$$\boldsymbol{f}(\boldsymbol{x}) = (\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{x}), \ldots, \boldsymbol{w}_n^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{x})). \tag{7}$$

Moreover, to improve the generalization capability, the bias terms are added in each component as follows:

$$\boldsymbol{f}(\boldsymbol{x}) = (D_1(\boldsymbol{x}), \ldots, D_n(\boldsymbol{x})), \tag{8}$$

where $D_i(\boldsymbol{x})$ are given by (1). And, using (8), we resolve the optimization problem in one-against-all formulation as follows:

$$\begin{aligned} \min \quad & \tfrac{1}{2}\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{v}_i + \tfrac{C'}{2}\textstyle\sum_{j=1}^{M}\xi_{ij}^2 && (9)\\ \text{s.t.} \quad & y_{ij}(\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{x}_j) + b_i') \geq 1 - \xi_{ij} \quad \text{for} \quad j = 1, \ldots, M, && (10) \end{aligned}$$

where $\boldsymbol{v}_i$ and b_i' are the n-dimensional weighted vector and the bias term. Because $n \leq M$, (9) and (10) are solved in primal form. Thus, the separating hyper-planes in the subspace are determined during considering other separating hyper-planes in the feature space.

4 Experimental Results

We compared the generalization ability of the conventional and the proposed multi-class SVMs, using benchmark data sets [3], [5], [6], [7], [8] listed in Table 1 that shows the number of classes, inputs, training data, and test data. In the computational experiments, we use least square SVMs (LS-SVMs)[9], in which the inequality constraints in L2-SVMs are converted into equality constraint conditions as follows:

$$y_{ij}(\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{x}_j) + b_i') = 1 - \xi_{ij} \quad \text{for} \quad j = 1, \ldots, M. \tag{11}$$

In training LS-SVMs, a set of linear equations is solved instead of a quadratic programming problem.

4.1 Parameter Setting

In the following study, we normalized the input ranges into [0, 1]. For the conventional and proposed methods, we determined a kernel type, a kernel parameter

Table 1. Multi-class benchmark data sets

Data	Classes	Inputs	Training	Test
Iris	3	4	75	75
Numeral	10	12	810	820
Blood-cell	12	13	3097	3100
Thyroid	3	21	3772	3428

of the selected kernels, and a margin parameter C by five-fold cross-validation for each problem. For the multi-class SVMs using the proposed feature extraction, we determined one more margin parameter C' in (9). We selected a kernel type from linear kernel: $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}'$, polynomial kernel: $(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}'+1)^d$, and RBF kernel: $\exp(-\gamma||\boldsymbol{x}-\boldsymbol{x}'||^2)$. If we selected polynomial or RBF kernels, we selected d or γ from $\{2,3,4,5\}$ or $\{0.1,0.5,1,1.5,3,5,10,15,20,30,50,100,200\}$. And we selected C and C' from $\{0.1,1,5,10,50,100,500,10^3,5\times10^3,10^4\}$. Table 2 shows the selected kernel and parameters for SVMs using the conventional model selection. Here, "Pol." denote polynomial kernels. Table 3 shows the selected margin parameter C' for the multi-class SVMs using the proposed feature extraction.

Table 2. Determined the hyper-parameters by the conventional model selection

Data	kernels	d or γ	C
Iris	Pol.	$d=3$	500
Numeral	RBF	$\gamma=10$	10
Blood-cell	RBF	$\gamma=5$	10^3
Thyroid	Pol.	$d=2$	5×10^3

Table 3. Determined the margin parameter C' for the multi-class SVMs using the feature extraction

Data	C'
Iris	0.1
Numeral	0.1
Blood-cell	0.1
Thyroid	10

And, Table 4–7 show the selected type of kernels, the kernel parameters, and the margin parameters by the above procedure for SVMs using the proposed model selection. In these tables, i denote the separating hyper-plane's number. From these tables, we can confirm that the same combination of the hyper-parameters as that of other separating hyper-planes is not selected for each separating hyper-plane. So, the combination determined by the conventional model selection in Table 2 is not always optimal for each separating hyper-plane.

Table 4. Determined the hyper-parameters by the proposed model selection for Iris dataset

i	kernels	d or γ	C
1	linear	–	0.1
2	RBF	$\gamma=10$	500
3	Pol.	$d=2$	10

Table 5. Determined the hyper-parameters by the proposed model selection for Thyroid dataset

i	kernels	d or γ	C
1	Pol.	$d=2$	10^4
2	Pol.	$d=3$	10^3
3	Pol.	$d=2$	10^3

Table 6. Determined the hyper-parameters by the proposed model selection for Numeral dataset

i	kernels	d or γ	C
1	Pol.	$d = 2$	5
2	Pol.	$d = 3$	50
3	Pol.	$d = 2$	1
4	RBF	$\gamma = 0.1$	500
5	Pol.	$d = 4$	500
6	linear	–	1
7	Pol.	$d = 2$	1
8	linear	–	1
9	Pol.	$d = 4$	50
10	Pol.	$d = 2$	1

Table 7. Determined the hyper-parameters by the proposed model selection for Blood-cell dataset

i	kernels	d or γ	C
1	RBF	$\gamma = 100$	10
2	RBF	$\gamma = 100$	1
3	RBF	$\gamma = 30$	10
4	RBF	$\gamma = 100$	50
5	RBF	$\gamma = 15$	100
6	RBF	$\gamma = 200$	50
7	RBF	$\gamma = 15$	500
8	RBF	$\gamma = 50$	100
9	RBF	$\gamma = 20$	500
10	RBF	$\gamma = 30$	100
11	Pol.	$d = 4$	5
12	RBF	$\gamma = 200$	5

4.2 Performance Comparison

Table 8 shows the recognition rates of the test data sets. In this table, the best results of the recognition rates in each row of the data sets are shown in boldface. And, "conventional SVMs", "SVM-*model selection*", and "SVM-*feature extraction*" denote SVMs using the hyper-parameters determined by the conventional model selection, that by the proposed model selection, and SVMs using the proposed feature extraction.

For all datasets, "SVM-*model selection*" perform better than "conventional SVMs". For the datasets except for Numeral dataset, "SVM-*feature extraction*" perform the best among all the methods. But, for other problems, "SVM-*feature extraction*" performed about the same as "conventional SVMs". From Table 8, we can conclude that multi-class SVMs using the proposed methods performed better than that using the conventional method, and "SVM-*feature extraction*" performs the best among all the methods from standpoint of classification ability.

Table 8. Comparison of the recognition rates in percent

Data	conventional SVMs	SVMs-*model selection*	SVMs-*feature extraction*
Iris	93.3	96.0	**97.3**
Numeral	**99.3**	**99.3**	99.1
Blood-cell	94.7	94.8	**95.1**
Thyroid	94.4	95.4	**95.8**

5 Conclusions

In this paper, we proposed the algorithm of the model selection for each separating hyper-plane and the feature extraction using SVMs for multi-class SVMs. Because the hyper parameters are determined separately for each separating hyper-plane, these are optimal. And, in multi-class SVMs using feature extraction by SVMs, the separating hyper-plane can be determined during considering other separating hyper-planes.

According to the computer experiments using multi-class benchmark datasets, the proposed methods perform much better than the conventional method.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Number 25871033.

References

1. Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, New York (1998)
2. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, New York (1995)
3. Abe, S.: Support Vector Machines for Pattern Classification (Advances in Pattern Recognition). Springer, London (2010)
4. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
5. Kitamura, T., Takeuchi, S., Abe, S., Fukui, K.: Subspace-based Support Vector Machines for Pattern Classification. Neural Networks 22(5-6), 558–567 (2009)
6. Kitamura, T., Takeuchi, S., Abe, S.: Feature Selection and Fast Training of Subspace Based Support Vector Machines. In: International Joint Conference on Neural Networks (IJCNN 2010), pp. 1967–1972 (2010)
7. Rätsch, G., Onda, T., Müller, K.R.: Soft Margins for AdaBoost. Machine Learning 42(3), 287–320 (2001)
8. `http://archive.ics.uci.edu/ml`
9. Suykens, J.A.K., Vandewalle, J.: Least Squares Support Vector Machine Classifiers. Neural Processing Letters 9(3), 293–300 (1999)

Restricted Boltzmann Machine with Adaptive Local Hidden Units

Binbin Cao[1], Jianmin Li[1], Jun Wu[2], and Bo Zhang[1]

[1] State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), and Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China*
caobinbin@live.com, {lijianmin,dcszb}@mail.tsinghua.edu.cn
[2] School of Electronics and Information, Northwestern Polytechnical University. Xi'an 710072, China
junwu@nwpu.edu.cn

Abstract. Deep belief network (DBN) shows the ability to learn hierarchical feature representation from image datasets which mimics the hierarchical organization of the mammal visual cortex. DBN is composed of a stack of Restricted Boltzmann Machines (RBM) which serves as feature extractors. A number of variants of RBM have been proposed to learn feature representations similar to gabor filters. They require extracting small image patches first. As images vary among different datasets, it is preferable to learn the patch size or a proper region of interest. We propose a variant of RBM with adaptive local hidden units (ALRBM) by adding a distance function to the connection weights between visible and hidden units. Experiments on hand-written digits and human faces show that our algorithm has the ability to learn region-based local feature representations adapting to the content of the images automatically.

1 Introduction

Feature representations have great impact on the performance of machine learning algorithms. Good feature representations tend to narrow down the semantic gap between raw data and human understanding. In the past years, much effort has been made on designing effective feature extraction methods. Design feature is labor-intensive and requires prior knowledge on dataset which is domain specific. So it is very desirable to build an unsupervised feature learning algorithm which can capture the information underlying the data automatically.

Fortunately, neuroscience researchers have provided insight into the principles of the visual system of mammals [1,2] which lead to new ideas for designing systems to deliver effective feature representations. The visual cortex of mammals

* This work was supported by the National Basic Research Program (973 Program) of China (Grant Nos. 2013CB329403 and 2012CB316301), National Natural Science Foundation of China (Grant Nos. 61273023, 91120011, 61103062), Beijing Natural Science Foundation (Grant No. 4132046), Tsinghua University Initiative Scientific Research Program No. 20121088071 and Tsinghua National Laboratory for Information Science and Technology (TNList) Cross-discipline Foundation.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 307–314, 2013.

which is responsible for processing the visual image consists of several secondary visual areas called V1, V2, V3, V4 and V5/MT that form a cortical hierarchy. Among those machine learning algorithms, Deep Belief Network (DBN) [3] is a popular deep learning approach which mimics the hierarchical framework of the mammal visual cortex. DBN can be viewed as a stack of simple learning modules, each of which is a Restricted Boltzmann Machine (RBM) that consists of a layer of visible units representing the data and a layer of hidden units learning to capture the higher-order correlations underlying the data. DBNs have been successfully adopted to various applications including hand-written digit recognition [3,4], face recognition [5] and object classification [6].

In visual information processing system, good feature representations often have two remarkable properties: locality and sparsity. Local and sparse feature representation is more robust and produces better generalization ability[7,8,9].

Several ways of incorporating sparsity into RBM have been proposed. Ranzato et al. [10] first incorporated sparsity into RBM by a so-called sparsifying logistic in 2006. Later they proposed a variant [11] through assigning a Student-t prior to the coders. Lee et al [12] developed a sparse RBM by adding a regularization term that penalizes a derivation of the expected activation of the hidden units from a fixed level. Their algorithms could learn localized, oriented edge filters similar to the gabor functions which are recognized as being able to model V1 cell receptive fields from hand-written digits and small natural image patches.

Locality is first incorporated into DBN by Convolutional Deep Belief Network (CDBN) [13]. In CDBN, features are extracted from convolutions of small filters with the whole image and are sent up to the next layer by probabilistic max-pooling. The size of the filter is hand-crafted. Because it is relative to the resolution and content of images, setting up a proper filter size is non-trivial with different datasets.

In this paper, we propose a different method to incorporate locality into RBM directly. We have developed a variant of RBM in which the connection weights between visible and hidden units are relative to the spatial distance between themselves. The region of interest and the connection weights can be learned simultaneously using the contrastive divergence learning rule. Experiments on hand-written digits and human faces show that our algorithm can learn region-based local feature representation.

2 Restricted Boltzmann Machine (RBM)

An RBM is a bipartite undirected graphical model with a set of hidden units $\boldsymbol{h}$, a set of visible units $\boldsymbol{v}$, and symmetric connection weights between these two layers represented by a weight matrix W. Suppose that we want to model k dimensional binary data using RBM with n binary hidden units, the negative log probability of any state in the RBM is given by the following energy function:

$$-\log P(\boldsymbol{v}, \boldsymbol{h}) = E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{i,j} v_i w_{ij} h_j - \sum_j b_j h_j - \sum_i c_i v_i \ . \qquad (1)$$

Here, h_j are hidden unit variables, v_i are visible unit variables, b_j and c_i are their biases. Because there are no inter connections between visible or hidden units, we can easily sample one layer from the other from the conditional probability distributions:

$$p(h_j = 1|\boldsymbol{v}) = \text{sigmoid}\left(\sum_i w_{ij} v_i + b_j\right) \ , \tag{2}$$

$$p(v_i = 1|\boldsymbol{h}) = \text{sigmoid}\left(\sum_j w_{ij} h_j + c_i\right) \ . \tag{3}$$

For training the parameters of the model, the objective is to maximize the log-likelihood of the data. The contrastive divergence learning algorithm gives an efficient approximation to the gradient of the log-likelihood [4]:

$$\Delta w_{ij} = \epsilon(< v_i h_j >_{\text{data}} - < v_i h_j >_{\text{recon}}) \ , \tag{4}$$

where ϵ is the learning rate, $< v_i h_j >_{\text{data}}$ is the frequency with which visible unit i and hidden unit j are on together driven by training data and $< v_i h_j >_{\text{recon}}$ is the corresponding frequency when the hidden units are being driven by reconstructed data. A similar learning rule can be used for the biases:

$$\Delta b_j = \epsilon(< h_j >_{\text{data}} - < h_j >_{\text{recon}}) \ , \tag{5}$$

$$\Delta c_i = \epsilon(< v_i >_{\text{data}} - < v_i >_{\text{recon}}) \ . \tag{6}$$

2.1 Gaussian Visible Units

For data such as natural images, binary unit is a very poor representation. One solution is to replace the binary visible units by linear units with independent Gaussian noise. The energy function then becomes:

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{i,j} \frac{v_i}{\sigma_i} w_{ij} h_j - \sum_j b_j h_j - \sum_i \frac{(v_i - a_i)^2}{2\sigma_i^2} \ . \tag{7}$$

If each component of the data is normalized to have zero mean and unit variance, we can use noise free reconstructions with the variance in (7) set to 1.

2.2 Sparse RBM

Lee et al. [12] proposed a variant of RBM with a regularization term that penalizes a deviation of the expected activation of the hidden units from a fixed level p. Thus, given a training set $\{v^{(1)}, \cdots, v^{(m)}\}$ comprising m examples, the optimization problem is as follows:

$$\text{minimize}_{w_{i,j}, c_i, b_j} -\sum_{l=1}^{m} \log \sum_{\boldsymbol{h}} P(\boldsymbol{v}^{(l)}, \boldsymbol{h}^{(l)}) + \lambda \sum_j |p - \frac{1}{m} \sum_{l=1}^{m} \mathbb{E}[h_j^{(l)} | \boldsymbol{v}^{(l)}]|^2 \ , \tag{8}$$

where $\mathbb{E}[\cdot]$ is the conditional expectation given the data, λ is a regularization constant, and p is a constant controlling the sparseness of the hidden units h_j. This can be learned by the contrastive divergence learning rule (4), followed by one step of gradient descent using the gradient of the regularization term on each iteration.

3 RBM with Adaptive Local Hidden Units (ALRBM)

In the visual system of mammals, the receptive field of a visual neuron is a specific region of visual space in which an appropriate stimulus can drive a response in the neuron[1,2]. Receptive fields have been mapped for all levels of the visual system from retinal ganglion cells to visual cortex cells. Sizes of receptive fields increase along the visual pathway. The hierarchy structure of receptive fields implies that a neuron in the higher level of the structure is mainly connected to neurons in a small region in the lower level.

To mimic the receptive fields in the visual system, we have proposed a variant of RBM with adaptive local hidden units, by adding a distance function $r(i, j, \boldsymbol{\theta}_j)$ to the connection weights between visible and hidden units. Each hidden unit h_j has its own $\boldsymbol{\theta}_j$. The energy function of ALRBM is given by:

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{i,j} v_i w_{ij} r(i, j, \boldsymbol{\theta}_j) h_j - \sum_j b_j h_j - \sum_i c_i v_i \ . \tag{9}$$

Based on the contrastive divergence learning rule, the original weight w_{ij} is updated by[1]:

$$\Delta w_{ij} = \epsilon \cdot r(i, j, \boldsymbol{\theta}_j) \cdot (< v_i h_j >_{\text{data}} - < v_i h_j >_{\text{recon}}) \ . \tag{10}$$

And the parameter $\boldsymbol{\theta}_j$ is updated by:

$$\Delta \boldsymbol{\theta}_j = \epsilon (< \sum_i v_i w_{ij} \frac{\partial r(i, j, \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j} h_j >_{\text{data}} - < \sum_i v_i w_{ij} \frac{\partial r(i, j, \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j} h_j >_{\text{recon}}) \ . \tag{11}$$

3.1 Distance Function in ALRBM

In visual information processing system, we arrange the visible units and hidden units in an RBM in 2-dimensional grids, see (Fig.1). Thus, each visible and hidden unit has x and y coordinates. Under this spatial configuration, a hidden unit will have locality property when it is strongly connected to nearby visible units. Here we assume the connection weights of a hidden unit take a 2-dimensional gaussian distribution. Thus, the distance function is given by:

$$r(i, j, \boldsymbol{\theta}_j) = \exp\left(-\frac{1}{2(1-\rho_j^2)} \left[\frac{\Delta x^2}{\sigma_{x_j}^2} + \frac{\Delta y^2}{\sigma_{y_j}^2} - \frac{2\rho_j \Delta x \Delta y}{\sigma_{x_j} \sigma_{y_j}} \right] \right) \ , \tag{12}$$

[1] Derivation here is omitted due to space constraints. It is similar to the derivation in [14].

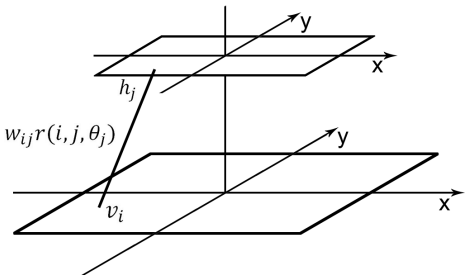


Fig. 1. Spatial configuration of ALRBM

where $\boldsymbol{\theta}_j = \{\rho_j, \sigma_{x_j}, \sigma_{y_j}\}$ are the parameters of the distance function of h_j, $\Delta x = x_{v_i} - x_{h_j}$ and $\Delta y = y_{v_i} - y_{h_j}$ are the distance along each axis between v_i and h_j.

4 Experiments

We train ALRBM, sparse RBM and original RBM on the MNIST database of hand-written digits [15] and CBCL face database #1 [16]. These algorithms are compared by the visualization of their extracted features and reconstruction errors.

4.1 Hand-Written Digits

The MNIST dataset contains hand-written digits from 0 to 9 in 28x28 gray-scale pixels. We have trained the original RBM, the sparse RBM and the ALRBM respectively using all of the training examples, each pixel is normalized to the unit interval. These networks have 784 binary visible units and 484 binary hidden units. We use the same $p = 0.02$ and $\lambda = 1/p$ for the sparseness of the hidden units as [12] in sparse RBM and ALRBM. σ_{x_j} and σ_{y_j} are initialized by 15 in ALRBM and ρ_j is initialized by 0.

4.2 Human Faces

Similar experiments are adopted on CBCL face database #1. All of the 2,429 faces in the training set are used. Each pixel is normalized to the unit interval, subtracted by the mean and divided by the standard deviation over all images. The networks have 361 gaussian visible units and 361 binary hidden units, σ_{x_j} and σ_{y_j} are initialized by 10, other parameters are the same as those mentioned in the previous experiment.

4.3 Discussions

Feature representation. Features learned by sparse RBM and ALRBM are visualized in Fig.2. Sparse RBM learns strokes from hand-written digits and face

prototypes from human faces. ALRBM learns smaller region based strokes from hand-written digits and region based facial features from human faces. In both experiments, ALRBM learns region based feature representations adapted to the content of the data. Further more, features learned by sparse RBM are cluttered in the background while those learned by ALRBM are much clearer.

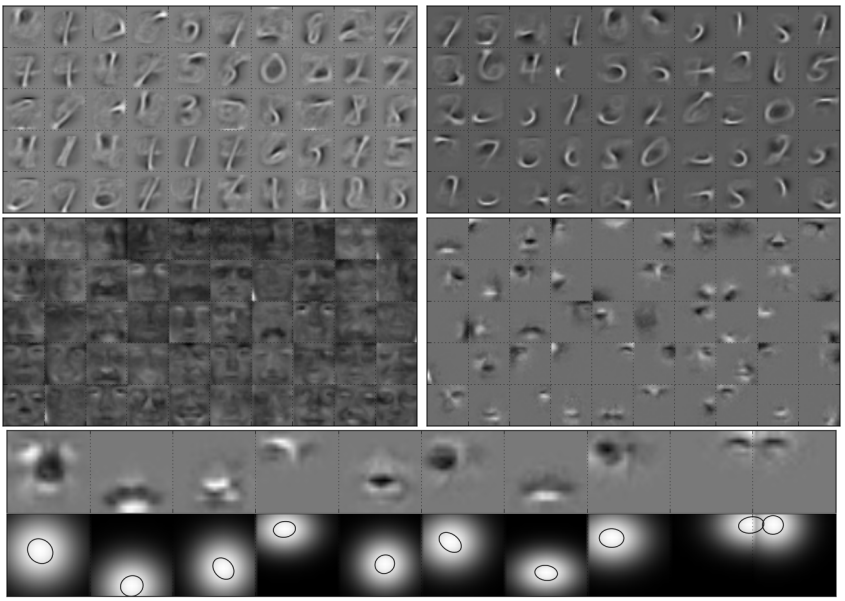


Fig. 2. Randomly selected features. Top-left: features learned by sparse RBM on MNIST. Top-right: features learned by ALRBM on MNIST. Middle-left: features learned by sparse RBM on CBCL faces. Middle-right: features learned by ALRBM on CBCL faces. Bottom: Some facial features and their corresponding region of interest selected from the features learned by ALRBM on CBCL faces.

Reconstruction power. In both experiments, reconstruction error of ALRBM is larger than that of sparse RBM (see Fig.3). Due to the local constraints introduced by the distance function, ALRBM tends to learn region-based local feature representation. Thus, global information cannot be well encoded. This can be easily overcome by merging sparse RBM and ALRBM together as complementation or by using ALRBM as building blocks to design a more complicated hierarchical feature representation framework similar to CDBN.

Initialization of σ_{x_j} and σ_{y_j}. ALRBM learns the regions of interest by first looking at larger areas and then contracting through iterations. Initialization of σ_{x_j} and σ_{y_j} should be large enough to learn meaningful local regions of interest. In our experiments, σ_{x_j} and σ_{y_j} are initialized to enable the hidden units to look at the whole image at first.

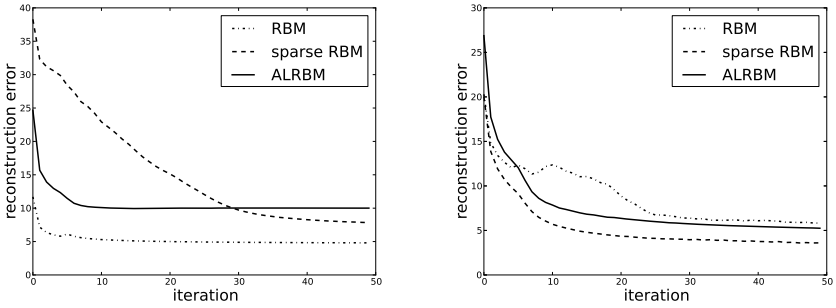


Fig. 3. Left: reconstruction error through iterations on MNIST; Right: reconstruction error through iterations on CBCL faces

Efficiency. The learning algorithm of ALRBM is similar to the original RBM except for additional parameters $\boldsymbol{\theta}_j$. Learning one element of $\boldsymbol{\theta}_j$ has the same complexity as learning w_{ij} which dominates the learning procedure of the original RBM. Having three elements in $\boldsymbol{\theta}_j$, ALRBM runs roughly 4 times slower than the original RBM in one iteration.

Convergence. Local constraints in ALRBM introduced by the distance function improve the stability and convergence of RBM. Figure.3 shows that ALRBM converges the fastest in both experiments.

5 Conclusions

We present a variant of RBM which encodes the locality property into the connection weights between the visible units and the hidden units by introducing a distance function. Experiments on hand-written digits and human faces show that our model has the very function to learn region-based local feature representation adapting to the content of the image data automatically.

References

1. Hubel, D.H., Wiesel, T.N.: Receptive fields and functional architecture of monkey striate cortex. The Journal of Physiology 195(1), 215–243 (1968)
2. De Valois, R.L., William Yund, E., Hepler, N.: The orientation and direction selectivity of cells in macaque visual cortex. Vision Research 22(5), 531–544 (1982)
3. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18(7), 1527–1554 (2006)
4. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
5. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML 2010), pp. 807–814 (2010)

6. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25, pp. 1106–1114 (2012)
7. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2169–2178. IEEE (2006)
8. Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y.: Robust face recognition via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(2), 210–227 (2009)
9. Coates, A., Ng, A.: The importance of encoding versus training with sparse coding and vector quantization. In: Proceedings of the 28th International Conference on Machine Learning (ICML 2011), pp. 921–928 (2011)
10. Ranzato, M., Poultney, C., LeCun, Y.: Efficient learning of sparse representations with an energy-based model. In: Advances in Neural Information Processing Systems, vol. 19, pp. 1137–1144 (2007)
11. Ranzato, M., Boureau, Y.-L., LeCun, Y.: Sparse feature learning for deep belief networks. In: Advances in Neural Information Processing Systems, vol. 20, pp. 1185–1192 (2007)
12. Lee, H., Ekanadham, C., Ng, A.: Sparse deep belief net model for visual area v2. In: Advances in Neural Information Processing Systems, pp. 873–880 (2007)
13. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 609–616. ACM (2009)
14. Ackley, D.H., Hinton, G.E., Sejnowski, T.J.: A learning algorithm for boltzmann machines. Cognitive Science 9(1), 147–169 (1985)
15. LeCun, Y., Cortes, C.: The mnist database of handwritten digits (1998)
16. Dataset: MIT center for biological and computation learning. CBCL face database #1, `http://www.ai.mit.edu/projects/cbcl`

Smooth Spatial Filter for Common Spatial Patterns

Xuan Li and Haixian Wang*

Research Center for Learning Science,
Southeast University, Nanjing, Jiangsu 210096, P.R. China
{xuanli,hxwang}@seu.edu.cn

Abstract. Common Spatial Patterns (CSPs) is a popular feature extraction algorithm for Brain-Computer Interface (BCI). However, the standard CSP spatial filters completely ignore the spatial information of EEG electrodes. To solve this problem, two smooth Regularized CSP (RCSP) algorithms are proposed in this paper, which are Spatially RCSP with a Gaussian Prior (GSRCSP) and Spatially RCSP with a Feature-Associations Modeling Matrix (MSRCSP) respectively. Then these algorithms are compared with the standard CSP and Spatially RCSP (SRCSP), an existing smooth CSP, in an experiment on EEG data from three publicly available data sets from BCI competition. Results show that GSRCSP outperforms other algorithms in classification accuracy and MSRCSP needs least training time. Besides, the spatial filters obtained by GSRCSP and MSRCSP are smoother than the standard CSP and SRCSP and are more interpretable neuro-physiologically.

Keywords: Brain computer interface (BCI), common spatial pattern (CSP), smooth, regularization, spatial information.

1 Introduction

Brain-Computer Interface (BCI) system aims at transforming the brain activity signals into different computer commands to control some applications designed for certain people. However, the input of BCI is always raw EEG signals with a lot of noise. To discriminate these signals effectively, features critical for classifying different mental states should be extracted. Common Spatial Patterns (CSPs) algorithm is such an effective feature extraction method for EEG classification in BCI.

However, as a spatial filters, CSP ignores the spatial information of EEG electrodes completely. From a neuro-physiological point of view, neighboring brain cells tend to function similarly, so neighboring electrodes should measure similar brain activity signals. Thus, we can expect that neighboring channels of the spatial filter should have similar weights (i.e. smooth spatial filter). This motivation inspires the improvement of the CSP to obtain smooth spatial filters.

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 315–322, 2013.

Lotte et al have proposed a Spatially Regularized CSP (SRCSP) to achieve this goal by using the spatial information as a priori knowledge [1]. Despite better results than the standard CSP, it needs quite a long time in learning spatial filter, which a good BCI should avoid. [2] and [3] also use spatial information to design smooth algorithms for MEG and fMRI data respectively, which inspires our proposal of new smooth CSP algorithms for EEG data. In this paper, two Regularized CSP (RCSP) algorithms have been proposed. They are Spatially RCSP with a Gaussian Prior (GSRCSP) and Spatially RCSP with a Feature-Associations Modeling Matrix (MSRCSP) respectively.

2 Method

2.1 Standard CSP

CSP aims at obtaining spatial filters that maximize the differences between two classes of EEG signals. Features extracted by these filters can help achieve optimal results for EEG classification in BCI. Formally, a popular method [4] is to extremize the following function:

$$J(w) = \frac{w^{\mathrm{T}} X_1^{\mathrm{T}} X_1 w}{w^{\mathrm{T}} X_2^{\mathrm{T}} X_2 w} = \frac{w^{\mathrm{T}} C_1 w}{w^{\mathrm{T}} C_2 w} \tag{1}$$

where X_i is an $S \times N$ data matrix for class i (with S as the number of samples and N as the number of channels), T represents transpose and C_i is the spatial covariance matrix from class i ($i \in [1, 2]$). Lagrange multiplier method can be used to solve this optimization problem subject to the constraint $w^{\mathrm{T}} C_2 w = 1$. Then the problem turns into the following function :

$$C_2^{-1} C_1 w = \lambda w \tag{2}$$

It is a standard eigenvalue problem.To get optimal results, usually we choose the first and last f eigenvectors of $M = C_2^{-1} C_1$ as the spatial filters w to extract features, which correspond to its largest and lowest eigenvalues respectively. After projecting the bandpass-filtered EEG signal variance onto w, the logarithm of it is in fact the features used to discriminate these signals.

2.2 Spatially RCSP with a Gaussian Prior

In order to obtain smooth spatial filter, based on the neuro-physiological knowledge mentioned before, we use the spatial location as a priori to regularize the standard CSP and design two smooth CSP algorithms. The first algorithm we propose is a GSRCSP, which adds spatial information in the form of Gaussian prior to regularize the standard CSP. The Gaussian prior is as follows:

$$Prior(w) \propto \exp\left(-\frac{1}{2} w^{\mathrm{T}} Q^{-1} w\right) \tag{3}$$

where Q is a positive-definite covariance matrix. A Gaussian prior allows us to model the correlations between weights in the matrix Q [2]. Thus the distances of EEG electrodes is used to determine the correlations between corresponding weights directly and encode them into CSP. In other words, weights closed to each others will be assigned high correlation, thereby ensuring the smoothness. Specifically, the following function defines Q:

$$Q(i,j) = \exp\left(-\frac{1}{2}\frac{\|v_i - v_j\|^2}{\sigma^2}\right) \tag{4}$$

where σ is a parameter determining the spatial smoothness and v_i is the 3-D spatial location vector of the ith electrode. To add the prior into the standard CSP, we take the logarithm of the prior and turn it into a penalty form $P(w)$, thus it becomes:

$$Prior(w) \propto \left(-\frac{1}{2}w^{\mathrm{T}}Q^{-1}w\right) \quad \text{and} \quad P(w) = w^{\mathrm{T}}Q^{-1}w \tag{5}$$

From a statistical point, the more the filters satisfy the prior, the larger $Prior(w)$ will be, correspondingly, penalty $P(w)$ will be lower. Then adding this penalty into the objective function of standard CSP, it becomes:

$$J_{p_1}(w) = \frac{w^{\mathrm{T}}C_1w}{w^{\mathrm{T}}C_2w + \alpha P(w)} \quad \text{and} \quad J_{p_2}(w) = \frac{w^{\mathrm{T}}C_2w}{w^{\mathrm{T}}C_1w + \alpha P(w)} \tag{6}$$

where α is a regularization parameter. To obtain the w, we need maximize $J_{p_1}(w)$ and minimize $P(w)$ meanwhile. Using the Lagrange multiplier method, it becomes the problem of solving the eigenvectors of $M_1 = (C_2 + \alpha Q^{-1})^{-1}C_1$ and $M_2 = (C_1 + \alpha Q^{-1})^{-1}C_2$ corresponding to the largest eigenvalues. To avoid twice matrix inversion and ensure the accuracy of calculation, they can be transformed into the following functions by doing some basic matrix operations:

$$M_1 = Q(C_2Q + \alpha I)^{-1}C_1 \quad \text{and} \quad M_2 = Q(C_1Q + \alpha I)^{-1}C_2 \tag{7}$$

where I is the identity matrix. GSRCSP will encourage the spatial filter to satisfy the correlation structure, so the neighbouring electrodes tend to have similar weights, thereby ensuring the spatial smoothness.

2.3 Spatially RCSP with a Feature-Associations Modeling Matrix

The other smooth CSP algorithm we propose is a MSRCSP. MSRCSP designs a matrix to model the associations between features (i.e. EEG electrodes) and uses this matrix to regularize CSP. Different from GSRCSP, MSRCSP uses a generalized ridge penalty as in [3] to include the spatial information, as follows:

$$P(w) = \|\Gamma w\|_2{}^2 \tag{8}$$

where Γ is the feature-association modeling matrix we need to define. For each feature, its associated features are selected according to the distances between

features. To be more specific, a parameter p is used to choose the first p nearest features as its associated features. Here we set parameter $p \in \{3, 4, 6\}$. Based on the neurophysiological knowledge, we should set every element in Γ properly to make sure that neighbouring features have similar weights when minimizing the penalty. For example, if feature F_a is associated with F_{b_1} and F_{b_2} and their corresponding weights are w_a, w_{b_1} and w_{b_2} respectively, then we can define Γ_a (the vector corresponding to F_a) like this:

$$\begin{array}{c} \Gamma_a = (\ 2\ ...{-1}\ {-1}\ ...) \\ \uparrow \quad \uparrow \quad \uparrow \\ a\ ...\ b1\ \ b2... \end{array} \tag{9}$$

It assumes that all features associated with one particular feature are equally important and the assumption can generally be met. Because, for each feature, the distances between the feature and its associated ones show little differences. $\Gamma = [\Gamma_1 ... \Gamma_N]^{\mathrm{T}}$ where N is the number of features (i.e. number of electrodes). For every Γ_i, we need to set every element in the way stated in (9). Similarly, the same method as stated in section 2.2 is used to add this penalty into the standard CSP. M_1 and M_2 turn into the functions below:

$$M_1 = (C_2 + \alpha \Gamma^{\mathrm{T}} \Gamma)^{-1} C_1 \ \text{ and } \ M_2 = (C_1 + \alpha \Gamma^{\mathrm{T}} \Gamma)^{-1} C_2 \tag{10}$$

By further observing, we can find that $P(w) = \sum_{i=1}^{N} (w_i - S_w)^2$, where S_w denotes the sum of weights corresponding to the features associated with the ith feature. When minimizing $P(w)$, it will force the weight of ith feature and its associated ones to be much closer, thus ensuring the smoothness of spatial filters.

2.4 Spatially Regularized CSP

SRCSP is an exsiting smooth CSP algorithm proposed by Lotte et al [1]. They add a Laplacian penalty term $P(w) = w^{\mathrm{T}} K w$ which includes the spatial information in a regularization matrix K as follows:

$$K = D_G - G \ \text{ with } \ G(i,j) = \exp\left(-\frac{1}{2}\frac{\|v_i - v_j\|^2}{r^2}\right), D_G(i,i) = \sum_i G(i,j) \tag{11}$$

where v_i is the 3-D coordinates of the ith electrod and r is a hyperparameter defining the smoothness. The penalty will encourage neighbouring electrodes to have similar weights to obtain smooth filters.

3 Experiment

To evaluate GSRCSP and MSRCSP, we compare them with standard CSP and SRCSP on EEG data of 17 subjects from 3 publicly available data sets from BCI competition in the experiment.

Table 1. Data Set

Data set	BCI competition III Data set IVa					BCI competition III Data set IIIa			BCI competition IV Data set IIa
Subject	A1	A2	A3	A4	A5	B1	B2	B3	C1-C9
Training set	168	224	84	56	28	90	60	60	144
Testing set	112	56	196	224	252	90	60	60	144
All trials	280					180	120		288
Electrodes	118					60			22
MI	Right hand and right foot					Left and right hand			Left and right hand

3.1 EEG Data Sets

Table 1 shows the detailed information about materials used in the experiment for all data sets. All EEG data used were collected when subjects were performing Motor Imagery (MI), which is an imagination of limbs movements [5]. For the purpose of our study, we only choose two kinds of MI for each data set. More precisely, right hand and right foot MI for Data set IVa, BCI competition III and left and right hand MI for the other two.

3.2 Preprocessing

For all trials, the time segment from 0.5s to 2.5s after visual cues of raw EEG signals was chosen and then bandpass filtered in 8-30 Hz [6], which include main bands for classification, using a fifth-order Butterworth filter. And we used 3 pairs of filters (i.e. f=3 as mentioned in section 2.1). For each subject, ten-fold cross-validation (CV) was used to find the best parameters for each algorithm and filters were learnt on training sets available using LDA classifier [7].

4 Results and Discussion

In the following part, four algorithms are compared and evaluated on three aspects of classification accuracy, smoothness and training time.

4.1 Classification Accuracy

Table 2 reports the results of classification accuracies on all data sets for each subject. It shows in both mean and median, GSRCSP does the best and outperforms CSP more than 3%. All three smooth algorithms outperform the standard CSP and have lower variances, which may suggest that smooth CSPs are more robust. Indeed, all subjects can be grouped into two categories: good subjects with higher accuracies and poor ones with lower accuracies. Further observation suggests that GSRCSP does best in both of the two groups (e.g. A2,B1,C2,C5) while MSRCSP has more advantages on the poor ones (e.g. A3,B2). A friedman

Table 2. Results

	BCI III									BCI IV								Overall		
	Data set IVa					Data set IIIa				Data set II										
Subject	A1	A2	A3	A4	A5	B1	B2	B3	C1	C2	C3	C4	C5	C6	C7	C8	C9	Mean	Median	Std
CSP	66.07	96.43	47.45	71.88	49.6	95.56	61.67	93.33	88.89	51.39	96.53	70.14	54.86	71.53	81.25	93.75	93.75	75.53	73.70	18.17
GSRCSP	72.32	100	57.14	82.14	79.76	97.78	56.67	93.33	88.89	57.64	96.53	63.89	62.5	65.28	81.25	93.75	90.97	78.81	80.51	15.70
MSRCSP	69.64	96.43	59.18	71.88	52.78	93.33	63.33	93.33	88.89	56.94	96.53	70.14	60.42	63.89	81.25	95.14	91. 67	76.75	74.31	15.93
SRCSP	72.32	96.43	58.16	72.32	87.30	96.67	53.33	93.33	88.89	52.78	96.53	70.14	56.25	65.97	81.94	95.14	91.67	78.19	80.07	16.4

test on GSRCSP, MSRCSP and SRCSP shows they have no significant differences ($p = 0.276$). However, it means these smooth CSPs are stable for all subjects and interchangeable when using CSP.

4.2 Smoothness

Fig.1 shows an example of filters obtained by four algorithms from six subjects. Overall, all smooth algorithms are smoother than the standard CSP since neighbouring weights of them are similar to each other. Furthermore, smooth filters can assign higher weights to the brain regions critical to MI classification [5]. So these filters can be more interpretable neurophysiologically.

A quantified method has also been used to assess the smoothness of these filters more precisely. For each weight, the sum of differences between it and its four nearest ones has been calculated. Then the summation of all these sums in each filter is used to measure the smoothness. Table 3 indicates that the order of

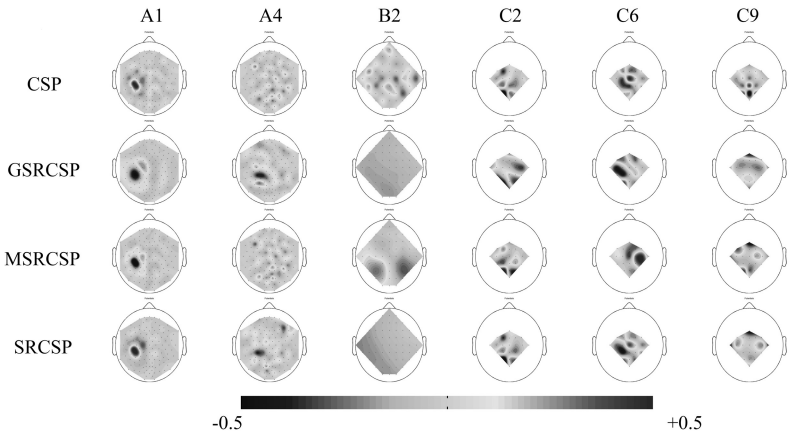


Fig. 1. Examples of weight vectors obtained by four algorithms (*CSP, GSRCSP, MSRCSP, SRCSP*) for subjects A1, A4 (*118 electrodes*), B2 (*60 electrodes*), C2, C6, C9 (*22 electrodes*)

Table 3. Smoothness

	A1	A2	A3	A4	A5	B1	B2	B3	C1	C2	C3	C4	C5	C6	C7	C8	C9	Mean	Median	Std
CSP	189.06	226.99	191.89	197.12	206.89	93.77	176.28	91.76	63.37	100.79	83.33	84.47	102.36	95.71	82.56	67.29	133.45	128.65	100.79	55.74
GSRCSP	93.27	148.01	61.45	142.99	69.02	67.47	11.16	91.76	63.37	71.36	72.12	31.19	88.50	86.53	82.59	62.25	70.80	77.28	71.36	33.03
MSRCSP	161.41	196.58	27.96	197.11	156.17	79.43	18.54	91.75	63.37	90.36	76.93	84.47	4.15	65.52	82.56	49.32	72.02	89.27	79.43	57.16
SRCSP	135.78	222.83	81.73	186.96	109.52	78.65	10.05	91.76	63.37	93.05	81.55	83.88	5.54	86.46	82.71	49.91	72.47	90.37	82.71	53.65

smoothness of filters from high to low is GSRCSP, MSRCSP, SRCSP and CSP. Besides, Friedman tests show that GSRCSP and MSRCSP both have significant differences from standard CSP ($p = 0.008$ and $p = 0.000$ respectively). So they are smoother than CSP substantially.

4.3 Training Time

Training time has also been compared among three smooth CSP algorithms. Table 4 suggests that for all data sets, the order of training time from short to long is MSRCSP, GSRCSP and SRCSP. A good BCI system should avoid tedious training procedure. Algorithms with short training time will show great advantages especially when the size of training trials is large.

Table 4. Traing Time

Training time mean-std /s	BCI competition III Data set IVa	Data set IIIa	BCI competition IV Data set IIa
GSRCSP	42.83-18-10	17.70-3.32	13.00-0.77
MSRCSP	21.57-9.10	8.82-1.75	6.95-0.34
SRCSP	57.68-24.55	24.87-5.10	18.66-0.96

5 Conclusion

In this paper, we propose GSRCSP and MSRCSP to get smooth spatial filters, by using spatial information to regularize the standard CSP. In the experiment, they are compared with CSP and SRCSP on three EEG data sets from two BCI competitions on three aspects of classification accuracy, smoothness and training time. Result shows that GSRCSP and MSRCSP outperform the standard CSP in classification accuracy and GSRCSP does the best. Although statistical tests show GSRCSP and MSRCSP have no significant differences from SRCSP, it means that they are interchangeable when using smooth CSPs. Besides, both GSRCSP and MSRCSP are smoother than CSP and SRCSP substantially in both the weight vector and the quantified comparison. Moreover, the two algorithms, especially MSRCSP, show great advantages on training time. In conclusion, GSRCSP and MSRCSP perform well in all three aspects. So they should be recommended in BCI designs especially when size of training trials is large.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China under Grant 61375118 and Grant 61075009, the Natural Science Foundation of Jiangsu Province under Grant BK2011595, and the Program for New Century Excellent Talents in University of China under Grant NCET-12-0115.

References

1. Lotte, F., Guan, C.T.: Spatially Regularized Common Spatial Patterns for EEG Classification. In: 20th International Conference on Pattern Recognition (ICPR), pp. 3712–3715. IEEE Press (2010)
2. de Brecht, M., Yamagishi, N.: Combining Sparseness and Smoothness Improves Classification Accuracy and Interpretability. NeuroImage 60, 1550–1561 (2012)
3. Ng, B., Abugharbieh, R.: Generalized Group Sparse Classifiers with Application in fMRI Brain Decoding. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1065–1071. IEEE Press (2011)
4. Lotte, F., Guan, C.T.: Regularizing Common Spatial Patterns to Improve BCI Designs: Unified Theory and New Algorithms. IEEE Trans. Biomed. Eng. 58(2), 355–362 (2011)
5. Pfurtscheller, G., Neuper, C.: Motor Imagery and Direct Brain-Computer Communication. Proc. IEEE 89(7), 1123–1134 (2001)
6. Ramoser, H., Muller-Gerking, J., Pfurtscheller, G.: Optimal Spatial Filtering of Single Trial EEG During Imagined Hand Movement. IEEE Trans. Rehabil. Eng. 8(4), 441–446 (2000)
7. Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., Arnaldi, B., et al.: A review of classification algorithms for EEG-based brain–computer interfaces. Neural Eng. 4, R1–R13 (2007)

Multiple Timescale Recurrent Neural Network with Slow Feature Analysis for Efficient Motion Recognition

Jihun Kim[1], Sungmoon Jeong[2], Zhibin Yu[1] , and Minho Lee[1,*]

[1] School of Electronics Engineering, Kyungpook National University
1370 Sankyuk-Dong, Puk-Gu, Taegu 702-701, South Korea
{ceuree,ahriman1985abr,mholee}@gmail.com
[2] School of Information Science, Japan Advanced Institute of Science and Technology,
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
jeongsm@jaist.ac.jp

Abstract. Multiple Timescale Recurrent Neural Network (MTRNN) model is a useful tool to learn and regenerate various kinds of action. In this paper, we use MTRNN as a dynamic model to analyze different human motions. Prediction error from dynamic model is used to classify different human actions. However, it is difficult to fully cover the human actions depending on the speed using dynamic model. In order to overcome the limitation of dynamic model, we considered Slow Feature analysis (SFA) which is used to extract the unique slow features from human actions data. In order to make input training data, we obtain 3 kinds of human actions by using KINECT. 3 dimensional slow feature data is be extracted by using SFA and those SFA feature data are used as the input of MTRNN for classification. The experiment results show that our proposed model performs better than the traditional model.

Keywords: motion recognition, multiple timescale recurrent neural network, slow feature analysis.

1 Introduction

Recognizing human motion is an important aspect in human-machine interaction based applications such as entertainment and engineering. Feature selection of human action is crucial to human motion recognition and the development of video sensors based technologies, such as KINECT, are opening new ways to design advanced human computer interface systems [1].

Hidden Markov Model is a traditional dynamic model to analyze and recognize human motions [2]. Yamashita and Tani [3] proposed a neuro-dynamic model referred to as multiple timescale recurrent neural network (MTRNN) to provide another way to model dynamic signals. As an extension of recurrent neural network (RNN) [4], MTRNN has an important feature called "self-organization", which is a phenomenon that a global coherent structure appears in a system not by a central authority

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 323–330, 2013.

but by local interactions among elements of the system [5].This particular feature makes MTRNN suitable for dynamic motion recognition tasks.

Although, video based technology can capture the whole motion sequence, tracking still becomes difficult because of some unpredictable differences in human actions. Usually, the start point and the moving speed are not exactly same even if we try to perform same motion twice. Hence, it is difficult to analyze the human motions using MTRNN alone. In addition, using KINECT makes it even more difficult to accurately estimate the moving speed of each skeleton node in each frame because of the zigzag noise in KINECT based motion data. MTRNN predictions are based on current input data as well as history data. Since, variation in zigzag noise is too large, MTRNN, in most of the cases, fails to follow fast changes.

To solve this problem, we need an alternative model which can generate smooth and slow signals. Slow feature analysis (SFA) [6] appears to be a suitable model for this. It is a kind of high-order network [7] and can obtain "slow" properties. Moreover, SFA is a kind of unsupervised learning algorithm of input in time sequence and obtains robust characteristics. These characteristics can be eventually used to analyze human motion classification [8, 9].

The paper is organized as follows: We present the overview of our model and related background in section 2. The comparison results are shown in section 3. Finally in last section, based on our experiment results, we conclude that the proposed model with SFA has performs better than a single MTRNN model.

2 Proposed Model

2.1 Overview of Proposed Model

Fig.1 shows an overview of the proposed human action recognition model. In this model, input training data are obtained from KINECT including the movement of 6 skeleton (12 dimension data including x and y axis) nodes on right hand. Each action is recorded during 3 seconds with 30 frame/sec sampling time (90 frames). Then, SFA is used as the feature extraction method for slow features from the KINECT data. The output of SFA, which only includes 3 dimensional data, is used as the input data of MTRNN.

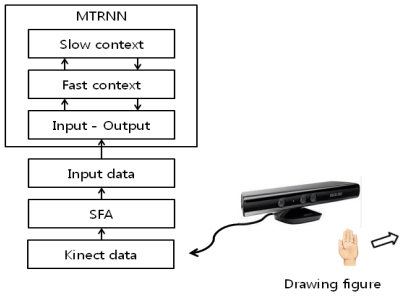


Fig. 1. The flow chart of proposed human action recognition model

The prediction error of MTRNN is used for motion recognition. Yamashita and Tani [3] used initial states for different actions. We also used 3 kinds of initial states for 3 different motions. Minimum prediction error between target signal and predicted signal depends on the selection of correct initial state. Therefore, to recognize human actions by finding minimum prediction error according to different initial states, we need to check each initial state and record the prediction error for a test motion signal

2.2 Slow Feature Analysis(SFA)

SFA is an unsupervised algorithm to learn nonlinear functions that extract slowly varying signals from time series. In this model, SFA can extract slow features from a dynamic signal which can be used as an input of MTRNN. Considering the feature of SFA, we expect SFA to extract similar features for the same class and different features for two different kinds of motions.

In SFA, input signal is given that $\mathrm{x(t)} = [x_1(t), \cdots, x_I(t)]^T$ in I-dimensional space. SFA finds out a set of input-output functions $\mathrm{g(x)} = [g_1(t), \cdots, g_J(t)]^T$, such that each output function is given by following equation:

$$y_j(t) \coloneqq g_j(\mathrm{x}(t)) \tag{1}$$

such that for each $\mathrm{j} \in \{1, \cdots, \mathrm{J}\}$

$$\Delta y_j(t) = \langle \dot{y}_j^2 \rangle_t \text{ is minimal} \tag{2}$$

under the constraints

$$\langle y_j \rangle = 0 \quad \text{(zero mean)}, \tag{3}$$

$$\langle y_j^2 \rangle = 1 \quad \text{(unit variance)}, \tag{4}$$

$$\forall j' < j\text{: } \langle y_{j'} y_j \rangle \quad \text{(decorrelation)}, \tag{5}$$

$\langle \cdot \rangle$ means averaging over time and $\dot{y}$ means derivative of y. Eq. (2) means temporal variation of the output signal. Constraints Eq. (3) and (4) normalize all output signals to a common scale. Constraints Eq. (5) means that different output signal carry different information.

The algorithm of SFA includes several steps.

1. Input signal normalization by following equation:

$$x_i(t) = (\tilde{x}_i(t) - \langle \tilde{x}_i \rangle) / \sqrt{(\langle (\tilde{x}_i - \langle \tilde{x}_i \rangle)\text{^}2 \rangle)} \tag{6}$$

2. Nonlinear expansion: Expansion of input signal use quadratic form for which following is the equation

$$\tilde{z}(t) \coloneqq \tilde{h}\big(x(t)\big) = [x_1(t), \cdots, x_I(t), x_1(t)x_1(t), x_1(t)x_2(t), \cdots, x_I(t)x_I(t)] \tag{7}$$

$\tilde{\mathrm{h}}\mathrm{(x)}$ is nonlinear function and $\tilde{\mathrm{z}}\mathrm{(t)}$ is expanded signal.

3. Sphering: Normalize the expanded signal $\tilde{z}(t)$ to transform z(t)

$$z(t) \coloneqq S(\tilde{z}(t) - \langle \tilde{z}(t) \rangle) \tag{8}$$

S is sphering matrix that normalizes expanded signal.

4. Principal component analysis: Apply PCA to matrix $\langle \dot{z}\dot{z}^T \rangle$. Get j eigenvectors with lowest eigenvalues λ_j and normalized weight vectors

$$\langle \dot{z}\dot{z}^T \rangle w_j = \ddot{e}_j w_j \text{ with } \ddot{e}_1 \leq \ddot{e}_2 \leq \cdots \leq \ddot{e}_j \tag{9}$$

Thus, input-output function is

$$g(x) \coloneqq [g_1(x), \cdots, g_j(x)]^T \text{ with } g_j(x) \coloneqq w_j^T h(x) \tag{10}$$

and the output signal

$$y(t) \coloneqq g(x(t)) \tag{11}$$

2.3 Multiple Timescale Recurrent Neural Network (MTRNN)

MTRNN model is based on continuous time recurrent neural network (CTRNN) model[10]. The neurons of slow and fast context layer have different time scales. MTRNN has three groups of neural units; input-output units, fast context units and slow context units. Input-output units contain 64 units to accept input information 3 dimensional data from SFA) using topology preserving map(TPM)[11].The number of fast and slow context units are set as 70and 30, respectively.

The membrane potential of these neurons is modeled by the conventional firing rate model, which is calculated by following linear differential equation:

$$\hat{o}_i \left(\frac{du_{i,t}}{dt} \right) = -u_{i,t} + \sum_j w_{ij} x_{j,t} \tag{12}$$

where $u_{i,t}$ is the membrane potential of each i-th neural unit at time step t and $x_{j,t}$ is the neural state of the j-th unit at time step t, and w_{ij} is a synaptic weight from the j-th unit to the i-th unit. The time constant τ is defined as the decay rate of a unit's membrane potential. If the τ value is large, units change slowly in time scale because the internal state potential is strongly affected by the history of unit's potential. On the other hand, if the τ value is small, the effect of history of unit's potential is also small. The activity of fast context units with small time constant (τ = 2) changes quickly, whereas the activity of slow context units with a large time constant (τ = 40) changes slowly. The updating of $u_{i,t}$ values is done by the numerical approximation of following equation:

$$u_{i,t+1} = \left(1 - \frac{1}{\hat{o}_i}\right) u_{i,t} + \frac{1}{\hat{o}_i} \sum_{j \in N} w_{ij} x_{j,t} \tag{13}$$

The activation of the i-th unit at time t is determined by the following equation:

$$y_{i,t} = \begin{cases} \frac{\exp(u_{i,t})}{\sum_{j \in Z} \exp(u_{i,t})} & if\ i \in Z \\ f(u_{i,t}) & otherwise \end{cases} \tag{14}$$

where Z is a set of output units that correspond to motion data. The softmax activation function is applied only to the output units, and not to the context units. Activation values of the context units are calculated by the function f which is a conventional unipolar sigmoid function$f(\mathrm{x}) = 1/(1 + e^{-x})$.

The MTRNN is trained to obtain the optimal connective weights by minimizing the learning error E. The error function E was defined by the Kullback-Leibler divergence, as shown in following equation:

$$\mathrm{E} = \sum_i \sum_{i \in O} y^*_{i,t} \log (y^*_{i,t} / y_{i,t}) \tag{15}$$

where $y^*_{i,t}$ is the desired activation value of the output neuron at time t, O is a set of output units, and $y_{i,t}$ is the activation value of the output neuron with the current connective weight. A conventional back propagation through time (BPTT) algorithm [12] was used to train the model. Through iterative calculation of the BPTT, the values of the connective weights reach their optimal values in the sense that the errors between a teaching sequence and an output sequence is minimized.

2.4 MTRNN with SFA

Suppose we get a group of N dimensional data obtained by KINECT in time step t called $\mathrm{X(t)} = [x_1(t), \cdots x_n(t), \ldots, x_N(t)]^T$. SFA will create a nonlinear mapping function and exchange the N dimensional input data$\mathrm{X(t)}$ to M dimensional output data $\mathrm{X'(t)} = [x'_1(t), \cdots x'_m(t), \ldots, x'_M(t)]^T$ by using the principles mentioned in section 2.2:

$$\mathrm{X'(t)} = SFA\{\mathrm{X(t)}\} \tag{16}$$

$\mathrm{X'(t)}$, which is the output of SFA is used as the input of TPM located in MTRNN:

$$p_{i,t} = \frac{\exp\left(-\frac{||k_i - \mathrm{X\prime(t)}||^2}{\acute{o}}\right)}{\sum_{j \in Z} \exp\left(-\frac{||k_j - \mathrm{X\prime(t)}||^2}{\acute{o}}\right)} \tag{17}$$

where$p_{i,t}$ is the activation value of TPM in time step t and will be used as $x_{j,t}$ in Eq.(12), $i \in Z$ means the i-th node of TPM, $k_i = \{k_{i,1}(t), \cdots k_{i,m}(t), \ldots, k_{i,M}(t)\}$ is the reference vectors set of TPM.

By combining the two models, slow and smooth data can be obtained as the input data of MTRNN using SFA.

3 Result

3.1 Result of SFA

Fig. 2 shows 3 kinds of signals marked by 3 colors. We obtained these results using minimum eigenvalue from SFA. As we mentioned in section 2.2, we need SFA to extract similar signals for the same kinds of motions. In Figs. 2 (a)-(c), there are 5 curves marked by different colors. Each curve is obtained by a pattern of human arm

motions such as circle, star and square. Obviously, the difference among 5 different curves is very small as compared to Figs. 2 (d)-(f) using KINECT data directly. On the other hand, there is a large difference between different kinds of motions as shown in Figs. 2 (a),(b) and (c). The zigzag noise is also reduced by using SFA. Thus, we can conclude that SFA is efficient to extract unique features from a kind of motion.

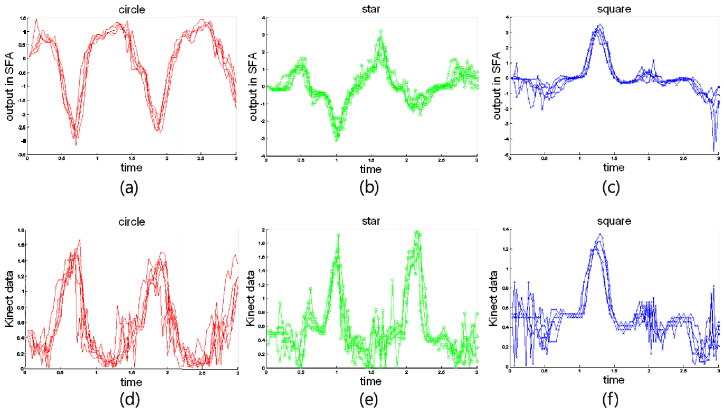


Fig. 2. Comparison between KINECT data and SFA feature data from 3 types of human action such as drawing circle, star and square. (a) circle(MTRNN+SFA), (b) star (MTRNN+SFA), (c) square (MTRNN+SFA), (d) circle (MTRNN), (e) star (MTRNN),and (f) square (MTRNN).

3.2 Prediction Error of Training

We also compare the prediction error between the conventional MTRNN method and the proposed model using MTRNN with SFA. The prediction error is defined as the difference between the output at the time step t and the input at the time step t+1 for training. Lower error means better performance. The mean square error (MSE) in input-output layer, which is the average square error per neuron per step over all teaching sequences, is used to evaluate the prediction accuracy. Fig. 3 shows the MSE changing over time for 2 different models using MTRNN only and MTRNN with SFA. The red curve corresponding to SFA model obtains smaller error than the model using MTRNN directly. The MSE of MTRNN using SFA data decreased below 0.002 after 2,000 iterations while MTRNN with KINECT data are still higher than 0.003. These results indicate that the learning speed of MTRNN with SFA is faster than the conventional MTRNN model.

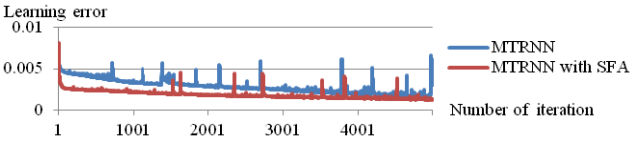


Fig. 3. Comparison of prediction error in 2 different models

3.3 Result of Recognition

The motion recognition result is shown in Table 1. It is clear that our proposed model achieves better recognition results as compared to the conventional MTRNN model. The star and square motions are perfectly recognized by our proposed model. Although some of the circle motions are wrongly classified as star motion in our model, it's much better than the performance of the MTRNN model.

Table 1. PERFORMANCE OF RECOGNITION

Motion type	Performance of using MTRNN	Performance of using MTRNN with SFA
circle	13%	73%
star	75%	100%
square	36%	100%

4 Conclusion

In this paper, we proposed a novel motion recognition model based on SFA and MTRNN. Our experiment results show that the SFA is able to extract slow unique features for different motions. These slow signals can be easily learned in MTRNN for human motion recognition using KINECT.

As our future work, we are trying to develop a KINECT based dance classification system including more complex human motion patterns. Also, we are applying the proposed model to recognize human intention by analyzing human gesture sequences.

Acknowledgments. This work was supported by the Industrial Strategic Technology Development Program(10044009), Development of a self-improving bidirectional sustainable HRI technology for 95% of successful responses with understanding user's complex emotion and transactional intent through continuous interactions) funded by the Ministry of Knowledge Economy(MKE, Korea) and was supported by the Converging Research Center Program through the Ministry of Science, ICT and Future Planning, Korea (2013K000333).

References

1. Cruz, L., Djalma, L., Luiz, V.: KINECT and RGBD Images. In: 25th SIBGRAPI Conference Challenges and Applications Graphics, Patterns and Images Tutorials (2012)
2. Joslin, C., El-Sawah, A., Chen, Q., Georganas, N.: Dynamic Gesture Recognition. In: Instrumentation and Measurement Technology Conference (2005)
3. Yamashita, Y., Tani, J.: Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment. PLoS Computational Biology 4(11) (2008)
4. Cleeremans, A., Servan-Schreiber, D., McClelland, J.L.: Finite state automata and simple recurrent networks. Neural Computation 1(3), 372–381 (1989)

5. Hinoshita, W., Arie, H., Tani, J., Okuno, H.G., Ogata, T.: Emergence of hierarchical structure mirroring linguistic composition in a recurrent neural network. Neural Networks 24(4), 311–320 (2011)
6. Wiskott, L., Sejnowski, T.J.: Slow feature analysis: Unsupervised learning of invariances. Neural Computation 14(4), 715–770 (2002)
7. Kosmatopoulos, E.B., Polycarpou, M.M., Christodoulou, M.A., Ioannou, P.A.: High-order neural network structures for identification of dynamical systems. IEEE Transactions on Neural Networks 6(2), 422–431 (1995)
8. Wiskott, L., Sejnowski, T.J.: Slow feature analysis: Unsupervised learning of invariances. Neural Computation 14(4), 715–770 (2002)
9. Koch, P., Konen, W., Hein, K.: Gesture recognition on few training data using Slow Feature Analysis and parametric bootstrap. In: The International Joint Conference on Neural Networks (2010)
10. Bruske, J., Sommer, G.: Dynamic Cell Structure Learns Perfectly Topology Preserving Map. Neural Computation 7(4), 845–865 (1995)
11. Saarinen, J., Kohonen, T.: Self-organized formation of colour maps in a model cortex. Perception 14(6), 711–719 (1985)
12. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: Parallel Distributed Processing, Explorations in the Microstructure of Cognition, 1st edn. MIT Press, Cambridge (1986)

A Comparison between Artificial Bee Colony and Particle Swarm Optimization Algorithms for Protein Structure Prediction Problem

Zakaria N.M. Alqattan and Rosni Abdullah

School of Computer Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
zqttan2@yahoo.com, rosni@cs.usm.my

Abstract. Protein Structure Prediction (PSP) is a well known problem for Bioinformatics scientists. It was considered as a NP-hard problem. Swarm Intelligence is a branch of evolutionary algorithm, is commonly used for PSP problem. The Artificial Bees Colony (ABC) optimization algorithm is inspired from the honey bees food foraging behavior and the Particle Swarm Optimization (PSO) algorithm which also simulate the process of the birds' foraging behavior are both used to solve the PSP problem. This paper investigates the performance of the two algorithms when being applied on an experimental short sequence protein called Met-enkaphlin in order to predict its 3D structure. The results illustrates clearly the power of the PSO search strategy and outperforms the ABC in terms of Time, Avg.NFE and success rate values by 70%, 73%, 3.6% respectively. However, the ABC results were more stable than the PSO in terms of Std.dev values, by 74%.

Keywords: Protein structure, Prediction, Artificial Bees Colony, Particle Swarm Optimization.

1 Introduction

Protein Structure prediction is one of the interesting concepts in the Bioinformatics field, as the importance of the proteins to the humanity. Proteins are involved in performing specific functions in the human's body. Their function and performance properties depend upon the three dimensional (tertiary) structures they have. Each protein is constructed from a sequence of amino acids connected to each other in a long chain. There are 20 different amino acids that combine together to build the protein's sequence, each of which may appears more than once in the same sequence [1]. According to [2], each protein may have between 20 to 40000 amino acids and most of the proteins have around hundreds of amino acids. According to the type and the number of the amino acids in the sequence, the protein will fold in special shape in the cell and take its three dimension structure.

Moreover; by knowing the 3D structure of the protein the scientists are able to determine its biological function. One of the methods that have been used to

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 331–340, 2013.

determine the protein's 3D structure is by using the experimental tools, such as NMR and X-ray Crystallography. Using these tools enabled the scientists to understand the biological function of the proteins as well as what type of designed drugs can be effective on them. However, using these experimental methods is very costly and time consuming [2-6]. In this paper, we investigate the performance of the two algorithms (ABC & PSO) on the PSP problem through a comparative study by using different parameter settings.

The rest of the paper is organized as follow: Section 2 explaining protein structure. Section 3 and 4, an overview of ABC and PSO algorithms. Section 5 describing the methodology for PSP used in this work. Results implementation of the proposed approach can be seen in Section 6. The discussion and future work are in Section 7, Conclusion and future work are discussed in Section 8.

2 Protein Structure

In the protein sequence, each amino acid consists of two main parts: main chain or backbone and side chain or R chain. The connected amino acids by peptide bonds generate what is called main chain or backbone from the generated side chains (R group). The main chain consist of a carbon alpha (C_a) which is bonded with an amino group (NH2), hydrogen atom (H) and carboxylic acid group (COOH). The R group or side chain is also connected to the carbon alpha Ca as shown in Fig 1. The carboxyl group of one amino acid is tending to join with the amino group of another by the formation called peptide bonds. Each of the joined amino acids ends is still unconnected, which is the amino group of the first one and the carboxyl group of the end once. This enables the sequence to be connected with other amino acids from the free sides [1].

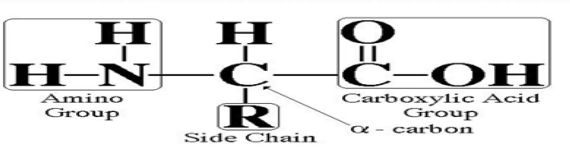


Fig. 1. Amino acid
Source: http://www.thefoodadvicecentre.co.uk/reference/protein

The representation of the protein is very important for the prediction operation. Since the peptide main chain bonds are effectively present the backbone by their connection, and the representation of the C_a atom along the protein sequence; the peptide ponds have the freedom to rotate around themselves. According to [1] "Each peptide unit can rotate around two bonds: the C_a-Ć and the N-C_a bonds, and by convention the angle of rotation around the N-C_a bond is called *Phi* (ϕ) and the angle around the C_a-Ć bond from the same C_a atom is called *Psi* (ψ)", Fig. 2.

Finally; since each amino acid in the protein sequence has only two main angles (dihedral angles) of rotating freedom which presents its conformation; these angles

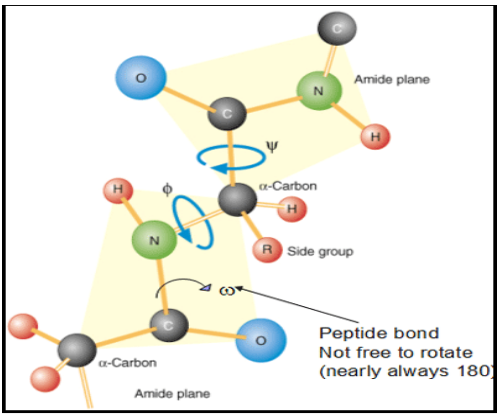


Fig. 2. The Phi (ϕ) and Psi (ψ) dihedral angles
Source: http://www.molecularsciences.org/book/export/html/128.

have the main responsibility to represent the whole main chain of the polypeptide in the protein sequence, which establish the final protein 3D structure [1].

In this study, Artificial Bee Colony (ABC) which is a foraging honey bee feature and particle Swarm Optimization (PSO) algorithms will be used in order to formulate the search algorithm for best protein conformation results.

3 Artificial Bee Colony Overview

In the ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. Each employed bee is sent to only one food source. In each cycle of the search, the ABC conducts three steps: send the employed bees to the food sources and calculate their amount of nectar. Then the onlooker bees are sent to measure the nectar of the food sources after getting the information about the location of the promising food sources. The scout bee will be determined after the food source is exhausted by its employed bee and onlooker bee.

In ABC algorithm each food source position represents a possible solution for the optimization problem. Each solution of the optimization problem is associated with fitness value. The ABC generates N population size of random solutions in the initial stage. which means that each solution represent a position of one food source and denoted as x_{ij}, where i represents the particular solution (i=1,2,…,N) and each solution is a D-dimension vector, and j represents its dimension (j=1,2,…,D).
The onlooker bee chooses a food source depending on the probability value Pi associated with the food source. Probability value for each food source is calculated by following equation (1).

According to the information gathered so far from the neighbor around $\Theta\, i$ the onlooker make a comparison whether to select that food source or its neighbor.

$$p_i = \frac{F(\theta_i)}{\sum_{i=1}^{N} F(\theta_i)}. \quad (1)$$

The position of the chosen neighbor is calculated by the following equation (2):

$$\theta_i(j+1) = \theta_{ij} + \beta(\theta_{ij} - \theta_{kj}), \quad (2)$$

Where i represents the particular food source position (i= 1,2,…,N), k represent the neighbor's randomly chosen position (k= 1,2,…,N), the k value should be different from the i value . $\beta$is a random number between [-1, 1] used to estimate the neighbor food sources around x_{ij}. As the ABC search becomes close to the global optimum, the search diameter around the x_{ij} shrinks. If the fitness of the new solution (food source) is better, it takes the place of the old one in the memory. Otherwise, the old position will remain. After a number of ABC specified search cycles the food source with unimproved fitness will be abandoned, and the scout bee will be sent to find a new food source based on the equation (3). Then the food source found by the scout bee will take place of the abandoned one.

$$\mathrm{x}_{\mathrm{ij}} = \mathrm{x}_{\mathrm{ij}} + rand[0,1](x_{j\max} - x_{j\min}). \quad (3)$$

4 Particle Swarm Optimization

The Particle Swarm Optimization (PSO) is a swarm intelligence algorithm inspired from the natural behavior of Swarms such as fish schooling or bird flocking in nature to avoid predators, seek food and mates, temperature.etc. Swarm intelligence is a technique based on simple proceeding units interactions (agents interactions), being used for problem solving processes. The inspiration of the Heppner and Grenander (1990) where they studied natures flocks of birds, schools of fish and swarms of insects; it was the initial ideas to Eberhart and kennedy (1995) to combine the cognitive abilities with social interaction [7].

In this algorithm, every single solution represents an individual which expresses a particle. Each individual is characterized by two parameters location and velocity. The location represents the solution of that particular individual. Each solution also has a corresponding fitness value. Thus, the goal is to find the solution of the best fitness value which indicates the best location found by any particle before. The most important parts of the algorithm are the evaluation equation and the comparison among the locations that the particles have gone through. In each loop, for an individual (particle) if the current position is better than any locations that it has visited before, then the individual's location is exchanged by the current one. And the global best location is defined as the location of an individual with best fitness value at that loop. After the comparison, the location and the velocity of each single particle is changed by the following equation:

$$\mathrm{V}_{\mathrm{i,d}}^{\mathrm{t+1}} = \mathrm{w}^{\mathrm{t}} * \mathrm{V}_{\mathrm{i,d}}^{\mathrm{t}} + \mathrm{c}_1^{\mathrm{t}} * \mathrm{r}_1 * \left(\mathrm{pbest}_{\mathrm{i,d}}^{\mathrm{t}} - \mathrm{X}_{\mathrm{i,d}}^{\mathrm{t}}\right) + \mathrm{c}_2^{\mathrm{t}} * \mathrm{r}_2 * \left(\mathrm{gbest}_{\mathrm{i,d}}^{\mathrm{t}} - \mathrm{X}_{\mathrm{i,d}}^{\mathrm{t}}\right). \quad (4)$$

$$X_{i,d}^{t+1} = X_{i,d}^{t} + V_{i,d}^{t+1} \quad (5)$$

where w represents inertia weight, c_1 and c_2 are learning factors which determine the relative influence of cognitive (self-confidence) and social (swarm-confidence) components, respectively, r_1 and r_2 are independent random numbers uniformly distributed in the range[0,1]. $V_{i,d}^{t+1}$ and $X_{i,d}^{t}$ and $pbest_{i,d}^{t}$ are the velocity, position and the personal best of *i*th particle in *d*th dimension for the *t*th iteration, respectively. The $gbest_{i,d}$ is the *d*th dimension of best particle in the swarm for the *t*th iteration.

5 The Methodology for PSP

In order to evaluate the performance of the two algorithms on protein structure prediction, a complete program was built. According to [8] the PSP process in general contains three main phases: conformations creation, energy calculation and search algorithm implementation. Fig.3 illustrates the proposed search method for PSP problem.

Conformations Creation: in this phase, the food sources of the search algorithm are generated, and then a rotation function to rotate the structure by a random value of main chain (Phi (ϕ) and Psi (ψ)) and side chain angles degree was applied. In each rotation a new conformation is added to the food sources. The number of the food sources is fixed from the beginning by the parameters initialization.

Energy Calculation: for each conformation generated from the previous phase; an energy calculation function is applied. This function is used to calculate the energy of the conformation (generated protein structure), which reflects the inner interaction of the protein resident. This function is represented as the objective function for the search algorithm. It calculates the energy of each conformation produced during the search phase. The equation used for energy calculation function contains interactions of van der Waals (ΔG_{vdw}) and electrostatic (Columbic) (ΔG_{elec}) interactions solely [9]. The energy equation $\Delta G = \Delta G_{vdw} + \Delta G_{elec}$ is represented in (6).

$$\Delta G = \Delta G_{vdw} \sum_{i,j} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \right) + \Delta G_{elec} \sum_{i,j} \frac{q_i q_j}{\varepsilon(r_{ij}) r_{ij}} \quad (6)$$

Many energy functions have been proposed in the previous researches. programs called physical-based “force fields” like: CHARMM [10], AMBER [11], OPLS [12] and the earlier one is SMMP by [13] (ECEPP/2 and ECEPP/3), had shown a convenient and effective result when being applied for small peptide proteins ([3]; [5]; [14]). In this work, ECEPP/2 force fields function of the SMMP program will be used to calculate the energy of the conformations being produced.

Search algorithm implementation: A search algorithm is used in this phase. For each cycle of the search algorithm the conformations creation and energy calculation functions are called. The goal of the algorithm is to reach the optimal minimum

solution of the problem which is in this case (PSP problem) the conformation with the lowest free energy. This is based on the assumption that the proteins tend to fold into their native state (structure) when they are on lowest energy [15].

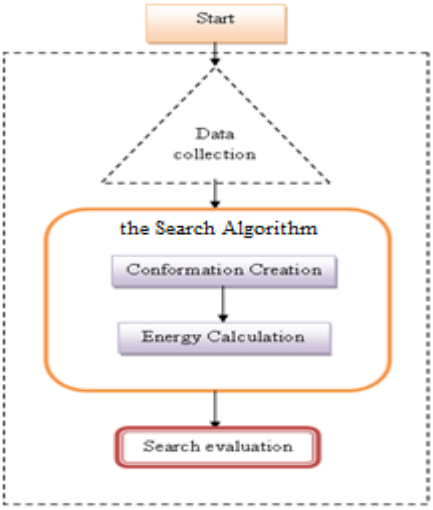


Fig. 3. The PSP search methodology[8]

Search Algorithm Implementation: A search algorithm is used in this phase. For each cycle of the search algorithm the conformations creation and energy calculation functions are called. The goal of the algorithm is to reach the optimal minimum solution of the problem which is in this case (PSP problem) the conformation with the lowest free energy. This is based on the assumption that the proteins tend to fold into their native state (structure) when they are on lowest energy [15].

6 Experimental Results and Discussion

In order to evaluate the performance of the two algorithms (ABC & PSO) for PSP problem, a Met-enkaphlin experimental protein was used. This is a short sequence protein with five amino acids resident. Each amino acid has two main chain angles (Phi (ϕ) and Psi (ψ)) and number of said chain angles. The number of the angles used that mainly influence the structure are 19 angles. Those angles are the D parameters of each solution in the search space. To intensively evaluate the performance of the two algorithms, various cases of different parameter settings were used: Table 1 for ABC settings and Table 2 for PSO settings. The proposed algorithms have control parameters: the colony size (SN= Employed bees + Onlooker bees), the Swarm population size (PN), the average number of function evaluations (NFE), "*limit*", inertia weight (ω), self-confidence (C_1) and swarm-confidence (C_2). We have compared the algorithms in terms of best , worst and mean fitness function values, standard deviation (SD), mean process time(S) and average number of function

evaluations (NFE). It should be noted here that the ABC algorithm stopping criteria is that of the number of cycles (or generations). However, in the present study we have considered NFE as stopping criteria for both algorithms. The maximum number of function evaluations is set to 10^5. In every case, a run was terminated when an accuracy of protein free energy $\leq$ -12.9101 (as reported in [5],[3]) was reached or when the maximum number of function evaluation was reached. The tests of each case are repeated for 30 runs and the average NFE is recorded. The experiment was implemented on a normal PC with ADM Athlon(tm) 7750 Dual-Core processor 2.70 GHz CPU and 4.00 GB RAM. The results of this test are shown by the Table 3 and 4.

Two types of information can be inferred the results. Firstly, the effects of the parameters settings, where each parameter has different effect on the algorithms' performance; secondly, it illustrates the advantages and the disadvantages of each algorithm especially when they had been applied on PSP problem.

As shown in table 3 the results of the ABC algorithm implicitly illustrate that the number of the bees and the limit parameter have the main influence on the performance of the algorithm. From Case 1 to 3 the effect of the parameter "*limit*" is clearly presented. *Limit* is a parameter which determines the food source to be abandoned. When the parameter is set to "100" the Std.dev and Ave.Time (s) has been decreased. It means that increasing the "*limit*" makes the algorithm more stable and to exclusively search the food source's neighbor before being abandoned. And from Case 1 to 5, the results show that increasing the number of the bees has not improved the performance of the algorithm when the *limit* parameter is set to low values; while Case 6 and 7 shown that increasing the bees' number with large value of *limit* has a very positive influence on the algorithm's performance.

While in table 4 the results of the PSO algorithm, regarding the parameters effects on the algorithm's performance, the results categorized a different reactions of PSO algorithm. In general, increasing the PN number has a good effect on the PSO performance but for a certain values only otherwise further increasing has no more positive effects (see Case1, 2 and Case5, 6). However, the global best location based strategy that the PSO uses, is the main rule for its effective search performance, otherwise there will be a disaster (see Case4). By giving parameter c2 larger value than the c1, the search will really further on the global information rather than the local or self-confidence information.

However, the research results in general, implicitly illustrate that in this cases the PSO algorithm performance has outperformed the ABC algorithm in terms of mean process-time, average NFE and success rate; But when taking into account the MAX and Std.dev evaluation parameters it is also clear that in contrast the ABC has some advantages regarding the algorithm's strategy of how to prevent itself from being trapped in the local minima and the algorithm's stability as well.

For further investigation on the ABC and PSO algorithms, the stopping condition of the ABC was set to 5000 iteration (regardless the number of evaluation) following some of the literature[17],[18]; while the stopping condition of the PSO was also set to be close to the number of evaluations done by the 5000 iteration of the ABC. The test was done on the best parameter settings cases produced by each algorithm so far. The results are shown in Table 5.

Table 1. ABC algorithm parameter settings

Cases	Employed Bee	Onlooker Bee	Limit
Case1	10	10	20
Case2	10	10	50
Case3	10	10	100
Case4	10	20	20
Case5	10	20	50
Case6	10	20	100
Case7	20	20	100

Table 2. PSO algorithm parameter settings

Cases	PN	c1	c2	w
Case1	20	1.49	2.0	0.4
Case2	25	1.49	2.0	0.4
Case3	30	1.49	2.0	0.4
Case4	20	1.49*	1.49*	0.7*
Case5	50	1.49	2.0	0.4
Case6	100	1.49	2.0	0.4

*based on the parameter settings used in [16]

Table 3. Results of applying ABC algorithm on Met-enkephalin using ECEEP/2 energy function*

Cases	Ave.	Max	Std.dev	Time(S)	Ave. NFE	Suc.rate %
Case1	-12.1428	-10.9101	0.86301	1315.56	77235.8	50
Case2	-12.5466	-10.9374	0.66403	1062.828	59651	70
Case3	**-12.5649**	**-11.1761**	**0.57797**	**1123.162**	**63914**	**63.33**
Case4	-12.3479	-10.9738	0.70802	1340.82	72688.7	46.66
Case5	-12.2102	-10.9101	0.79150	1379.668	76094.1	50
Case6	**-12.6368**	**-11.1761**	**0.57369**	**894.1145**	**50994.6**	**73.33**
Case7	**-12.7039**	**-11.49**	**0.49064**	**1019.673**	**55815.7**	**73.33**

*the Min value (optimal solution) is -12.9101 always. Time(s) is the average time

Table 4. Results of applying PSO algorithm on Met-enkephalin using ECEEP/2 energy function

Cases	Ave.	Max	Std.dev.	Time(S)	Ave. NFE	Suc.rate %
Case1	-12.2137	-9.8923	1.1873	719.3937	43143.3	73.33
Case2	**-12.6320**	**-11.49**	**0.7493**	**611.3442**	**35630.8**	**86.67**
Case3	-12.5373	-11.1761	0.7965	648.9	38529.0	80
Case4	-12.0624	-10.7945	0.8158	1547.652	86725.3	26.66
Case5	**-12.6716**	**-10.1592**	**0.6523**	**549.9918**	**33116.7**	**86.67**
Case6	-12.5219	-10.1592	0.8348	774.0623	42373.3	80

*the Min value (optimal solution) is -12.9101 always. Time(s) is the average time

Table 5. ABC and PSO comparison using 5000 iteration

Cases	Ave.	Max	Std.dev.	Time(S)	Ave. NFE	Suc.rate%
Case7/ ABC	-12.8803	-12.0181	0.1628	1573.699	93995.9	90
Case5/ PSO	-12.8150	-11.4797	0.3555	751.9739	43358.3	93.33

*the Min value (optimal solution) is -12.9101 always. Time(s) is the average time

Finally, it appears that the ABC algorithm's strategy of individual's movement using the equation (2) by choosing a random neighbor has shown its weakness especially for the PSP problem while the PSO strategy of individual movement using equation (4) was much better.

7 Conclusion

The experiments have shown the difference between the performance of the two algorithms ABC and PSO in term of the effects of the algorithms parameters initialization used and also the search strategies that the two algorithms used. The experiments were done on an experimental protein named Met-enkaphalin which is a short sequence protein of five amino acids residents. The number of the main chain and side chain rotatable angles is 18. The performance was interesting in terms of lowest energy found and acceptable in terms of time. For further research, a hybridization process between the two algorithms which combine the advantages of them in order to improve the search performance in an efficient and more stable manner.

Acknowledgments. Research reported here is pursued under the Fundamental Research Grant Scheme (FRGS) by Ministry of Higher Education (MOHE) for "Bio-Inspired Optimization Method for Feature Selection of Mass Spectrometry Analysis in Biomarker Identification of Ovarian Cancer" [203/PKOMP/6711268]. The authors also would like to thank Universiti Sains Malaysia (USM) for the support and facilities provided.

References

1. Branden, C., Tooze, J.: Introduction to protein structure. Garland, New York (1991)
2. Fidanova, S., Lirkov, I.: Ant colony system approach for protein folding. In: International Multiconference on Computer Science and Information Technology, IMCSIT 2008, pp. 887–891 (2008)
3. Abual-Rub, M.S., Al-Betar, M.A., Abdullah, R., Khader, A.T.: A hybrid harmony search algorithm for ab initio protein tertiary structure prediction. Network Modeling Analysis in Health Informatics and Bioinformatics 1, 69–85 (2012)
4. Bahamish, H.A.A., Abdullah, R., Salam, R.A.: Protein conformational search using bees algorithm. In: Second Asia International Conference on Modeling & Simulation, AICMS 2008, pp. 911–916 (2008)

5. Bahamish, H.A.A., Abdullah, R., Salam, R.A.: Protein tertiary structure prediction using artificial bee colony algorithm. In: Third Asia International Conference on Modelling & Simulation, AMS 2009, pp. 258–263 (2009)
6. Fonseca, R., Paluszewski, M., Winter, P.: Protein Structure Prediction Using Bee Colony Optimization Metaheuristic. Journal of Mathematical Modelling and Algorithms 9, 181–194 (2010)
7. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, MHS 1995, pp. 39–43 (1995)
8. Mahmood, Z.N., Mahmuddin, M., Mahmood, M.N.: Protein Tertiary Structure Prediction Based on Main Chain Angle Using a Hybrid Bees Colony Optimization Algorithm. International Journal of Modern Physics: Conference Series 9, 143–156 (2012)
9. Cui, Y., Chen, R.S., Wong, W.H.: Protein folding simulation with genetic algorithm and supersecondary structure constraints. Proteins: Structure, Function, and Bioinformatics 31, 247–257 (1998)
10. Brooks, B.R., Bruccoleri, R.E., Olafson, B.D., Swaminathan, S., Karplus, M.: CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. Journal of Computational Chemistry 4, 187–217 (1983)
11. Weiner, S.J., Kollman, P.A., Case, D.A., Singh, U.C., Ghio, C., Alagona, G., Profeta, S., Weiner, P.: A new force field for molecular mechanical simulation of nucleic acids and proteins. Journal of the American Chemical Society 106, 765–784 (1984)
12. Jorgensen, W.L., Tirado-Rives, J.: The OPLS [optimized potentials for liquid simulations] potential functions for proteins, energy minimizations for crystals of cyclic peptides and crambin. Journal of the American Chemical Society 110, 1657–1666 (1988)
13. Eisenmenger, F., Hansmann, U.H., Hayryan, S., Hu, C.-K.: [SMMP] A modern package for simulation of proteins. Computer Physics Communications 138, 192–212 (2001)
14. Hernández, L.G.P., Vázquez, K.R., Juárez, R.G.: Estimation of 3d protein structure by means of parallel particle swarm optimization. In: 2010 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2010)
15. Alm, E., Baker, D.: Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures. Proceedings of the National Academy of Sciences 96, 11305–11310 (1999)
16. Sharma, T.K., Pant, M., Bhardwaj, T.: PSO ingrained Artificial Bee Colony algorithm for solving continuous optimization problems. In: 2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), pp. 108–112 (2011)

Incremental Learning on a Budget and Its Application to Power Electronics

Koichiro Yamauchi, Yusuke Kondo, Akinari Maeda,
Kiyotaka Nakano, and Akihisa Kato

Department of Information Science, Chubu University
yamauchi@cs.chubu.ac.jp
http://sakura.cs.chubu.ac.jp

Abstract. In this paper, we present an incremental learning method on a budget for embedded systems. We discuss its application for two power systems: a micro-converter for photovoltaic and a step down DC-DC-converter. This learning method is a variation of the general regression neural network but it is able to continue incremental learning on a bounded support set. The method basically learns new instances by adding new kernels. However, when the number of kernels reaches a predefined upper bound, the method selects the most effective learning option from several options: including replacing the most ineffective kernel with the new kernel, modifying of the parameters of existing kernels, and ignoring the new instance.

The proposed method is compared with other similar learning methods on a budget, which are based on kernel perceptron. Two examples of the application of the proposed method are demonstrated in power electronics. In these two examples, we show that the proposed system learns the properties of the control-objects during the services and realizes quick control.

Keywords: Incremental Learning on a budget, Kernel Method, micro-converter, photovoltaic, shadow-flicker, model-based control, DC-DC converter.

1 Introduction

These days, almost all of electric circuits are controlled by micro-computers. A microcomputer that can execute a learning algorithm will enable the creation of embedded systems that are apt for present day scenario, aiding in circumstances such as age deterioration of materials and environmental changes.

The learning algorithm should not only be light-weighted but also compact. Such learning algorithms have been proposed as the variations of kernel perceptrons[1–3]. These algorithms learn instances using a fixed number of support sets to suit them for use in embedded systems, which have a limited storage capacity. In our previous works, we had proposed the Limited General Regression Neural Network (LGRNN) [4]. According to the preliminary results reported

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 341–351, 2013.

by [5], LGRNN is superior to the other methods. This is chiefly because LGRNN executes the best learning option after considering the predicted error in each of available learning options. In this paper, we propose a slightly extended version of the previous version of LGRNN[5] to reduce the computational complexity of the learning algorithm.

In addition, we present two examples of LGRNN applications. The first example is a photovoltaic micro-converter that learns the properties of the photovoltaic while it is working, and controls the chopper circuits to maximize the solar power generated [5]. We have presented an extended version of our previous work [5], which operates well even under shadow flicker conditions. The second example is a simple step down DC-DC converter, which learns the property of the load while it is working. This converter reacts to the sudden changes in the input voltage.

The remainder of this paper is organized as follows. In section 2 LGRNN is explained and comparisons between its performance and that of other similar learning methods is presented. Section 3 describes the two examples of the application of LGRNN, a micro-converter for photovoltaic and a DC-DC converter. The conclusion is presented in Section 4.

2 Limited General Regression Neural Networks

The LGRNN[4] is an extended GRNN. GRNN normally allocates a new kernel on the memory to learn a new instance. The LGRNN, however, continues to learn new instances on a limited memory capacity.

In this paper, the LGRNN algorithm is described briefly. Detailed derivations of the equations are described in Ref[4]. However, in this paper, the algorithm is slightly improved to simplify the learning algorithm (See Eq(7)(9)).

Although LGRNN output function is almost the same as that of GRNN [6], we represent LGRNN output function using a vector in Hilbert space. Therefore, the output value $y(\boldsymbol{x})$ is

$$y(\boldsymbol{x}) = \frac{\langle f_t, K(\boldsymbol{x}, \cdot)\rangle}{\langle g_t, K(\boldsymbol{x}, \cdot)\rangle},\ f_t = \sum_{j \in I_t} w_j K(\boldsymbol{u}_j, \cdot),\quad g_t = \sum_{j \in I_t} R_j K(\boldsymbol{u}_j, \cdot), \tag{1}$$

where w_j and R_j denote the output connection strength and the number of learned samples of the j-th hidden unit. Let f_t, g_t be the functions after the t-th iterations. I_t denotes the size of support sets. We can regard each hidden unit as a Gaussian kernel function so that $\langle K(\boldsymbol{u}_j, \cdot), K(\boldsymbol{x}, \cdot)\rangle = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{u}_j\|^2}{2\sigma^2}\right)$. In the initial state, LGRNN has no kernels. If a new instance $(\boldsymbol{x}_t, y_t)$ is presented, LGRNN appends a new kernel to record this new instance. If the number of kernel reaches an upper bound, LGRNN essentially replaces the most ineffective kernel with a new kernel, whose center position is equivalent to the new current input vector.

Unfortunately, there are cases in which the replacement process destroys a part of the knowledge that is learned and the generalization capability is degraded.

To overcome this problem, when the upper bound for the number of kernels is reached, the LGRNN selects one of the four learning options according to the predicted error. The four learning options are explained in section 2.2. Moreover, section 2.1 explains how the most ineffective kernel is determined to be replaced with a new kernel.

2.1 Finding the Most Ineffective Kernel

The kernel, which is to be replaced with a new kernel, is determined by an approximated linear dependency. Let us assume that one of the vectors $K(\boldsymbol{u}_i \cdot)$ can be written as a linear combination of $K(\boldsymbol{u}_j, \cdot)$ $(j \neq i)$. This means that $K(\boldsymbol{u}_j, \cdot)$ is redundant in this function approximation.

Therefore, the system chooses the i-th hidden unit that has the smallest value δ_i:

$$\delta_i = \min_{\boldsymbol{a}_i} \left\| K(\boldsymbol{u}_i, \cdot) - \sum_{j \neq i} a_{ij} K(\boldsymbol{u}_j, \cdot) \right\|^2 . \tag{2}$$

The hidden unit having the minimum δ_i value is suitable for being relieved of its duty since the adverse effects from its substitution is minimal. The optimal value of $\boldsymbol{a}_i$ and δ_i are obtained from:

$$\boldsymbol{a}_i = K^{-1} k(\boldsymbol{u}_i), \quad \delta_i = \left\{ 1 - k^T(\boldsymbol{u}_i) \boldsymbol{a}_i \right\}, \tag{3}$$

where $[K]_{ij} = K(\boldsymbol{u}_i, \boldsymbol{u}_j)$ and $k(\boldsymbol{x}) = [K(\boldsymbol{u}_1, \boldsymbol{x}), \cdots, K(\boldsymbol{u}_{i-1}, \boldsymbol{x}), K(\boldsymbol{u}_{i+1}, \boldsymbol{x}), \cdots]^T$.

The system chooses the i-th kernel where $i = \arg\min_j \{\delta_j\}$, and projects the i-th kernel to the space spanned by $I_{t-1} - \{i\}$ according to $\boldsymbol{a}_i$ derived in Eq(3), where I is the support set. After the projection, a new kernel is appended, whose center is at the new instance.

The replacement process is represented by

$$\begin{aligned} f_t &= f_{t-1-i} + \tau_i w_i P_{t-1-i} K(\boldsymbol{u}_i, \cdot) + y_t K(\boldsymbol{x}_t, \cdot), \\ g_t &= g_{t-1-i} + \tau_i R_i P_{t-1-i} K(\boldsymbol{u}_i, \cdot) + K(\boldsymbol{x}_t, \cdot), \end{aligned} \tag{4}$$

where f_{t-1-i} and $P_{t-i} K(\boldsymbol{u}_i, \cdot)$ denote the function after pruning the i-th kernel and the projected vector, respectively. $\tau_i \in \{0, 1\}$ denotes the projection ratio. If $\tau_i = 0$, the projection process is not executed.

Similarly, there is a possibility that the learning is achieved without replacement. In this case, the new instance is projected to the space spanned by I_{t-1}.

$$f_t = f_{t-1} + \tau_a y_t P_{t-1} K(\boldsymbol{x}_t, \cdot), \quad g_t = g_{t-1} + \tau_a P_{t-1} K(\boldsymbol{x}_t, \cdot). \tag{5}$$

A similar technique, which uses an Approximated Linear Dependency (ALD), was presented by Xu et al.[7]. The authors applied this method to a kernel machine that learns the 'Q-value' for reinforcement learning. In their method, ALD is used to evaluate whether or not a new kernel should be added. A "Projectron" proposed by Francesco et al.[2] uses a similar technique as the Xu's model for

improving the kernel perceptron based classifier. If a new instance can be represented by a linear combination of the existing kernels, the Projectron projects the new instance by modifying the coefficients of all kernels, otherwise, it adds a new kernel for recording the instance.

Perceptron with Dynamic Memory (PDM) proposed by Wenwu He et al.[3] discussed a similar technique called the "decremental projection" to reduce the adverse effect of replacing a kernel. They applied the method on a kernel perceptron. However, their method applies the decremental projection even if important past memory is being destroyed.

Our method, on the other hand, determines whether each learning option is really applied according to the prediction of the effect of the substitution process beforehand. So, our method is better than PDM to retain crucial memory.

2.2 LGRNN Algorithm

The pseudo code of LGRNN learning algorithm is shown in Algorithm1.

Algorithm 1. Pseudo-code for LGRNN

Require: new learning sample $(\boldsymbol{x}_t, y_t)$, upper bound for # of RBFNN hidden units: B, the importance weight of new samples: N_{new} The previous functions f_{t-1}, g_{t-1}
  if J (# of hidden units) $+1 \leq B$ **then**
    $f_t = f_{t-1} + y_t k(\boldsymbol{x}_t, \cdot)$, $g_t = g_{t-1} + k(\boldsymbol{x}_t, \cdot)$,
  else
    {find the most ineffective hidden unit}
    Calculate δ_j $(j = 1, 2, \cdots, B)$ by using Eq(3).
    $i \leftarrow \arg\min_j \{\delta_j\}$
    {Choose the next action from the four options}
    Estimate the expected loss of the four options (Eq(7)(8)(9)(10)).
    set $\tau_p \in \{0, 1\}$ and $\tau_a \in \{0, 1\}$ according to the selected learning option.
    if $e_{substitute}$ or $e_{replace}$ is the minimum of all **then**
      $f_t = f_{t-1-i} + \tau_p w_i P_{t-1-i} K(\boldsymbol{u}_i, \cdot) + y_t K(\boldsymbol{x}_t, \cdot)$
      $g_t = g_{t-1-i} + \tau_p R_i P_{t-1-i} K(\boldsymbol{u}_i, \cdot) + K(\boldsymbol{x}_t, \cdot)$
    else
      $f_t = f_{t-1} + \tau_a y_t P_{t-1} K(\boldsymbol{x}_t, \cdot)$
      $g_t = g_{t-1} + \tau_a P_{t-1} K(\boldsymbol{x}_t, \cdot)$
    end if

  end if
  RETURN f_{t+1}, g_{t+1}

The early steps of the LGRNN algorithm is the same as that of the original GRNN. However, when the number of kernels reaches the upper bound, LGRNN selects the best learning option from the four options. The selection of the best learning option is achieved in accordance to the predicted errors observed after the application of each of the four learning options.

The predicted errors are calculated as follows. Let e_* denote the expected loss for a learning option. Then, e_* should be the sum of the predicted magnitude of forgetting old samples and the predicted error of new samples.

$$e_* \equiv \int_{\boldsymbol{x} \in old} \{y^{new}(\boldsymbol{x}) - y^{old}(\boldsymbol{x})\}^2 P(\boldsymbol{x}) d\boldsymbol{x} + N_{new}\{y_t - y^{new}(\boldsymbol{x}_t)\}^2 \quad (6)$$

where $P(\boldsymbol{x})$ and N_{new} are the probability distribution of input and the importance weight of the new instance, respectively.

The four learning options and their predicted losses are explained in the following section. To estimate $P(\boldsymbol{x})$ and the importance weight, each kernel also counts the number of learned samples, N_i. N_i is normally a natural number but is a real number in cases where the i-th unit substitutes other ineffective units. The LGRNN selects the learning option, which has the least expected loss.

- **Pruning with Substitution and Replacement:** This option sets the projection and append ratios as $(\tau_i, \tau_a) = (1, 0)$. The expected loss in this option is the sum of the loss due to the projection and pruning. Therefore,

$$e_{substitute} \equiv \sum_{j \neq i} \left\{ \frac{a_{ji}(w_i^* - w_j^*)}{R_j(t) + R_i(t)a_{ji}} \right\}^2 N_j + N_i(w^*_{Nearest(i)} - y_t)^2 \delta_i \quad (7)$$

  where i denotes the index of the most ineffective kernel and w_i^* is $w_i^* \equiv w_i/R_i$. N_i denotes the number of samples, recorded by the i-th kernel. The default number of N_i is one, but if the i-th kernel substitutes another kernel, N_i increases: $N_i := N_i + N_j|a_{ji}|/\{\sum_k |a_{jk}|\}$. $Nearest(i)$ denotes the nearest kernel to the i-th kernel. Note that the second term of Eq(7) is a simplified term over our previous work[5] without affecting its performance.
  However, if $R_j + R_i a_{ij} < 0$ for $j \neq i$, this option is passed over for the candidate R_j to prevent from being negative value.
- **Pruning with Replacement:** The "Pruning with Substitution and Replacement" option sometimes causes interference because of the projection process. A "Pruning with Replacement" option achieves the replacement process without the projection process so that $(\tau_i, \tau_a) = (0, 0)$. However, the number of learned samples for the i-th unit is added to $N_{Nearest(i)}$ before replacement, where the latter is the number of samples learned by the nearest kernel. $N_{Nearest(i)} := N_{Nearest(i)} + N_i$ The expected loss from this option is the loss due to pruning. Therefore,

$$e_{prune} \equiv (w^*_{Nearest(i)} - w_i^*)^2 N_i, \quad (8)$$

  After the replacement, N_i is reset to 1.
- **Modification:** This option sets the projection and append ratios as $(\tau_i, \tau_a) = (0, 1)$. The expected loss with this option is the sum of the losses due to the projection and pruning. Therefore,

$$e_{modify} \equiv \sum_j \left\{ \frac{a_{jnew}(y_{new} - w_j^*)}{R_j(t) + a_{jnew}} \right\}^2 N_j + N_{new}(w^*_{Nearest(New)} - y_t)^2 \delta_{new} \quad (9)$$

where $Nearest(New)$ denotes the nearest kernel to the new instance and $\delta_{new} = \|K(\boldsymbol{x}_t, \cdot) - P_{t-1}K(\boldsymbol{x}_t, \cdot)\|^2$. N_i is updated by $N_i := N_i + |a_{inew}| / \{\sum_j |a_{jnew}|\}$.

– **Ignore:** Sometimes, doing nothing is the best option. In this option, $(\tau_i, \tau_a) = (0, 0)$. The expected loss is the loss cased by doing nothing.

$$e_{ignore} \equiv N_{new}(y_t - y(\boldsymbol{x}_t))^2 \tag{10}$$

After the estimation procedures, the option with the least expected loss is selected for incremental learning.

2.3 Comparison with Other Methods

The proposed LGRNN, which is an extension of our previous work, was compared with other kernel perceptron based methods: Projectron++ [2], PDM [3] and the crisp-projection method for kernel perceptron[8]. The crisp-projection method is similar to LGRNN learning but it uses kernel perceptron as its output function. We checked the effect of using GRNN based output function by comparing LGRNN's performance with that of the crisp-projection method.

The procedure for this comparison is similar to that of our previous work [8]. These models were examined by using the servo, housing for regression stored in the UCI machine learning repository[9]. A benchmark test was repeated 50 times by changing the dataset sequence. The results were averaged over 50 trials and 95% confidence intervals were also estimated.

Fig 1 shows an example of the performances for servo and housing datasets. In this example, the number of kernels was restricted to 10. We can see that the LGRNN shows the smallest Mean Squared Error(MSE) of all in the early steps of the learning in serve dataset, but the MSE is closed to that of crisp-projection in the latter steps of the learning. In the case of housing dataset, LGRNN shows the smallest MSE of all.

As LGRNN chooses the best learning option from the four learning options including the ignore option, it can continue the learning without increasing the errors. Moreover, the output function of LGRNN Eq(1) reduces the errors if the input includes noise[10] making the generalization capability of LGRNN superior to that of the kernel perceptron based crisp-projection.

LGRNN performances were also compared with other algorithms for different upper bound on the number of kernels. Fig 2 shows the results in case of servo and housing datasets. In this figure, the mean square error had been examined after the learning of all learning samples. We can see that LGRNN is superior to PDM and Projectron even if the upper bound of the number of kernels is small.

3 Applications of LGRNN

LGRNN is useful to embed the learning engine into a micro-computer. It might be able to speed up the traditional adaptive systems such as feedback controller by using LGRNN to predict the solution directory. A LGRNN system integrated with traditional adaptive system needs to be designed to ensure that LGRNN learns correct solutions.

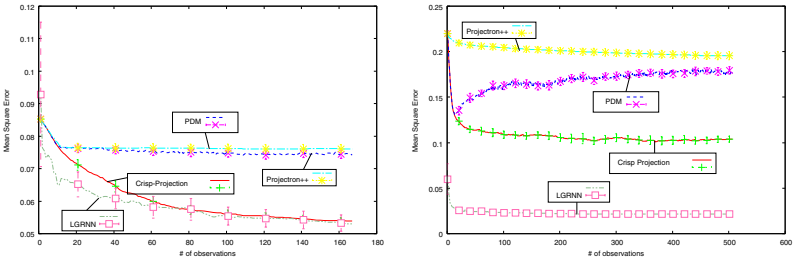


Fig. 1. Performances for servo(left) and housing(right) datasets. Upper bound on the number of kernels: 10, Vertical bar denotes 95% confidence interval

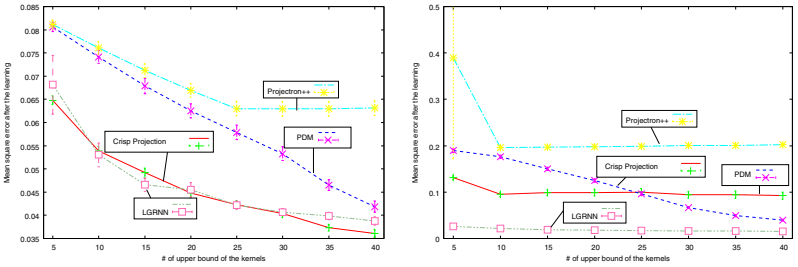


Fig. 2. The mean squared error versus the support set. The upper and lower figures are the results for servo(left) and housing(right) dataset, respectively.

3.1 Application for a Micro-converter of Photovoltaic

Solar panel is a current source, whose current strength depends on the solar radiation. Therefore, we have to set the output voltage of the solar panel correctly to obtain the maximum power. The voltage, which generates the maximum power is called "Maximum Power Point(MPP)" and is varies depending on solar radiation, and the temperature on the surface of the solar panel. Traditionally, MPP is tracked by the perturbation and observation (P&O) algorithm. However, the P&O method tracks MPP gradually, and its response is delayed in case of sudden changes in solar radiation. Various MPPT algorithms have been proposed [11] to overcome this problem.

In our previous work, we had developed a micro converter, which functioned as the maximum power point tracker for each solar panel using LGRNN [5]. This system consisted of a micro computer (H83069F) and an step up / down chopper. The micro computer executes a conventional P&O method, LGRNN and a PID controller for the chopper circuit (Fig. 3).

If irradiation is stable, the P&O yields the operating photovoltaic(PV) voltage. However, if the irradiation changes rapidly, the LGRNN yields the operating voltage according to the learned MPP.

Although the previous one[5] realizes quick responses to the sudden changes in the solar radiation, it fails to react to the shadow flicker quickly because

Algorithm 2. Pseudo code of proposed MPPT algorithm

solar irradiation S, tempeleture T, PV voltage V_{PV}, current I, and ResumeFlag.
if $V_{PV} < V_{threshold}$ **then**
  $\underline{ResumeFlag = 1}$; Save LGRNN parameters and shutdown.
end if
$\boldsymbol{x} = (S, T)^T$, $\Delta S := S - S_{previous}$, $P := I \times V_{PV}$
$\boldsymbol{u_{nearest}}$ = the nearest kernel center to $\boldsymbol{x}$
if $\underline{ResumeFlag == 1}$ or $|\Delta S| > \gamma$ and $\|\boldsymbol{x} - \boldsymbol{u_{nearest}}\|^2 < \theta$ **then**
  $\overline{V_{ref}}$ =LGRNN output y
  $ResumeFlag = 0$.
else
  if $P < P_{previous}$ **then**
    $\Delta V := -\Delta V$
  end if
  $V_{ref} := V_{ref} + \Delta V$
end if
$\overline{V_{ref}}$ = moving average of V_{ref} during the last 20 steps.
$\overline{\Delta V}$ = moving average of ΔV during the last 20 steps.
if $\overline{\Delta V} == 0$ **then**
  Make the LGRNN learn $(\boldsymbol{x}, \overline{V_{ref}})$
end if
$S_{previous} := S$
$P_{previous} := P$
RETURN V_{ref}

the micro-computer has to search the MPP again from its initial state after recovering from the short interruption due to shadow-flicker. The solar panels located near by a large wind turbine face many shadow flicker situations. In this paper, we extended the previous system[5] to realize an efficient power generation even under the shadow flicker conditions. This has been achieved by changing the load for the micro-converter from rechargeable-battery to a constant-current load (see Fig. 6). Therefore, the constant current condition makes the behaviors of series connected micro-converter stable. Moreover, the software that yields LGRNN output immediately after the micro-converters recover from shadow flicker has been charged.

The micro converter connected to a 50W silicon solar panel was tested using a 1500W halogen lamp. In this experiment, we created pseudo shadow flicker situations by letting the halogen lamp blink, and investigated the status of the micro converter under such a situation. An example of the results obtained is shown in Figure 4. It can be observed from this result that the proposed micro-converter reacts to the short shadow flicker corresponding to the sudden changes in solar radiation. However, if the duration of the shadowing was prolonged a power-down sequence of the micro-converter was observed. Even in the latter case, the modified MPPT algorithm enables the activation of LGRNN activate in the start-up timing so that the reference voltage was recovered.

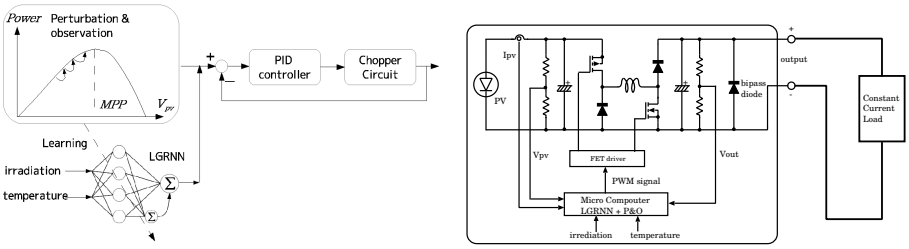


Fig. 3. MPPT method using LGRNN and P&O methods and its circuit configuration

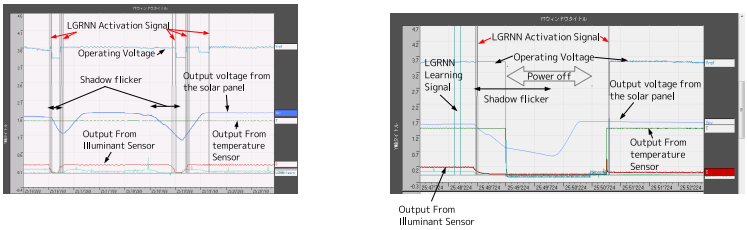


Fig. 4. An example of the behavior of the Micro converter under shadow flicker conditions: Short shadow flicker(left), Long shadow flicker(right). (Upper bound on the number of kernels: 20)

3.2 Application for a DC-DC Converter

We also developed a model-based controller using the LGRNN (See 5).

In this controller, the chopper circuit is normally controlled by PID controller. When the LGRNN is certain of the current input, the controller is switched from the PID controller to the LGRNN. However, when the controller is switched back to the PID controller, care must be taken to maintain stability. This is because, under normal circumstances, the PID controller outputs is not equivalent to that of LGRNN. Therefore, to ensure that stability maintained, the proposed controller resets the PID controller equals that of LGRNN.

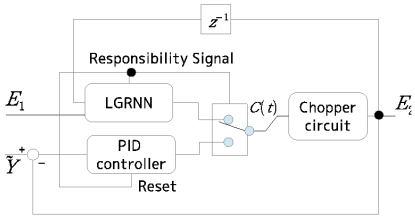


Fig. 5. Model-based controller for step down chopper circuit

The DC-DC converter was made to work on various input voltages (15V to 20V) to aid LGRNN learn. After that, the output voltage was checked by altering the input voltages.

Algorithm 3. Rough Pseudo code of the model-based control

input voltage $E_1(t)$, target voltage $\tilde{Y}$, output voltage $E_2(t-1)$
$\boldsymbol{x}_t = (E_1(t), E_2(t-1))^T$
if $\|\boldsymbol{x} - \boldsymbol{u}_{\boldsymbol{nearest}}\|^2 < \theta$ **then**
  $C(t)$ =LGRNN output y
  reset the PID controller.
else
  $C(t)$ = PID controller output.
end if
if $C(t)$ is converged **then**
  Make the LGRNN learn $(\boldsymbol{x}_t, C(t))$
end if
RETURN $C(t)$

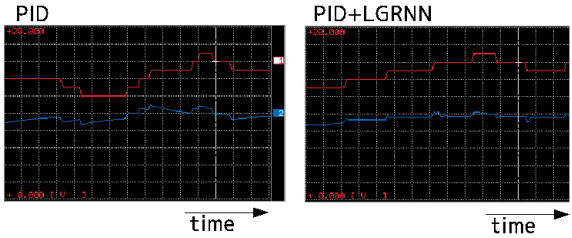


Fig. 6. Responses of DCDC converters: Only using PID controller(left), LGRNN together with PID controller(right). Red curve: Input voltage, Blue curve: Output voltage.(Upper bound on number of the kernels: 10)

Figure 6 shows the output voltage from the DC-DC converter for varying input voltages. The behavior of the DC-DC converter controlled by PID controller was also recorded for comparison. It is evident from the figure that the PID controller with LGRNN respond quickly to the changes in input voltage. Note that the converter controlling the chopper circuit was forced to perform in a slow pace to record the behavior clearly. The speed of the PID controller with LGRNN was about 5 times faster than that of PID controller individually.

4 Conclusion

Limited General Regression Neural Network(LGRNN) is presented in this paper. LGRNN continued the incremental learning using a fixed number of kernels. The comparison of LGRNN with the other kernel perceptron based learning methods provided that LGRNN is superior to the rest.

LGRNN was also applied to the micro converter of photovoltaic, which realized the Maximum Power Point Tracking(MPPT). In particular, the MPPT method in this paper was improved to realize the effective power generation even under the shadow flicker conditions. That is a condition which the solar panels located near wind turbines often face. In addition, we applied LGRNN to the

model-based controller of DC-DC converter. The results provided that LGRNN can speed up the feedback control.

Acknowledgment. This research has been partly supported by JST Adaptable and Seamless Technology Transfer Program through Target-driven R&D Exploratory Research AS221Z01499A, AS231Z00475B, Grant-in-Aid for Scientific Research(c) 12008012 and donation to this research by KYODO corporation.

References

1. Dekel, O., Shalev-Shwartz, S., Singer, Y.: The forgetron: A kernel-based perceptron on a budget. SIAM Journal on Computing (SICOMP) 37(5), 1342–1372 (2008)
2. Orabona, F., Keshet, J., Caputo, B.: The projectron: a bounded kernel-based perceptron. In: ICML 2008, pp. 720–727 (2008)
3. He, W., Wu, S.: A kernel-based perceptron with dynamic memory. Neural Networks 25, 105–113 (2011)
4. Yamauchi, K.: Pruning with replacement and automatic distance metric detection in limited general regression neural networks. In: Proceedings of International Joint Conference on Neural Networks, San Jose, California, USA, July 31-August 5, pp. 899–906. IEEE (2011)
5. Yamauchi, K.: Incremental learning on a budget and its application to quick maximum power point tracking of photovoltaic systems. In: The 6th International Conference on Soft Computing and Intelligent Systems, pp. 71–78 (November 2012)
6. Specht, D.F.: A general regression neural network. IEEE Transactions on Neural Networks 2(6), 568–576 (1991)
7. Xu, X., Hu, D., Lu, X.: Kernel-based least squares policy iteration for reinforcement learning. IEEE Transactions on Neural Networks 18(4), 973–992 (2007)
8. Yamauchi, K.: An importance weighted projection method for incremental learning under unstationary environments. In: IJCNN 2013: The International Joint Conference on Neural Networks 2013, pp. 506–514. IEEE (August 2013)
9. Bache, K., Lichman, M.: UCI machine learning repository (2013), `http://archive.ics.uci.edu/ml`
10. Webb, A.R.: Functional approximation by feed-forward networks: a least-squares approach to generalization. IEEE Transactions on Neural Networks 5(3), 363–371 (1994)
11. Esram, T., Chapman, P.L.: Comparison of photovoltaic array maximum power point tracking techniques. IEEE Transactions on Energy Conversion 22(2), 439–449 (2007)

Using Similarity between Paired Instances to Improve Multiple-Instance Learning via Embedded Instance Selection

Duzhou Zhang[1,2] and Xibin Cao[1,⋆]

[1] School of Astronautics, Harbin Institute of Technology
[2] National Key Laboratory for Space Intelligent Control Technology, China Academy of Space Technology
{dzzhang_hit,xbcao_hit}@163.com

Abstract. Multiple-instance Learning (MIL) copes with classification of sets of instances named bags, as opposed to the traditional view that aims at learning from single instances. Recently, several instance selection-based MIL algorithms have been presented to tackle the MIL problem. Multiple-Instance Learning via Embedded Instance Selection (MILES) is so far the most effective one among them, at least in our experiments. However, MILES regards all instances in the training set as initial instance prototypes, which leads to high complexity for both feature mapping and classifier learning. In this paper, we try to address this issue based on the similarity between paired instances within a bag. The main idea is choosing a pair of instances with the lowest similarity value from each bag and using all such pairs of instances as initial instance prototypes that are applied to MILES instead of the original set of initial instance prototypes. The evaluation on two benchmark datasets demonstrates that our approach can significantly improve the efficiency of MILES while maintaining or even strengthening its effectivenss.

Keywords: Multiple-instance learning, Instance selection, Similarity.

1 Introduction

Multiple-instance learning (MIL) is a variation on standard supervised learning, which was first introduced by Dietterich et al. when they were investigating the problem of drug activity prediction [1]. In this learning framework, training examples are bags of instances not single ones. Labels are associated with bags rather than instances in them. A bag is labeled as positive if it contains at least one positive instance; otherwise, labeled as negative. The aim of a MIL algorithm is to learn a classifier for predicting the labels of unseen bags.

The notion of bag together with the labeling protocol often make MIL more realistic than standard supervised learning for particular types of applications, such as drug activity prediction [1], stock selection [2], natural scene classification

⋆ Corresponding Author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 352–359, 2013.

[3], computer aided diagnosis [4], content-based image retrieval (CBIR) [5], as well as object detection [6,7] and tracking [8,9].

In recent years, several instance selection-based MIL (ISMIL) algorithms have been proposed, including DD-SVM [10], MILES [11], MILD [12] and MILIS [13], which will be described later in Sect. 2. MILES (Multiple-Instance Learning via Embedded Instance Selection) is the most effective one among all these algorithms, which has been demonstrated by our experimental studies on the MUSK and COREL datasets. However, MILES considers all instances in the training set as initial instance prototypes, which leads to high complexity for the subsequent feature mapping and classifier learning, especially for large-scale datasets. As we know, computational efficiency is an important issue for practical applications, so it is necessary to design an efficient approach to speed up the whole learning process of MILES while not sacrificing its generalization accuracy much. Following the above analysis, we know that the high computational complexity of MILES is mainly due to the fact that it uses all instances in the whole training set as initial instance prototypes. Inspired by this observation, we attempt to improve the efficiency of MILES from the perspective of instance pruning. In this paper, we propose an efficient instance pruning approach to address the above issue based on the similarity between paired instances within a bag. We call it **I**nstance **P**runing via **S**imilarity between **P**aired Instances (PSIP). The main idea is choosing a pair of instances with the lowest similarity value from each bag and using all such pairs of instances as initial instance prototypes, which are applied to MILES instead of the original set of initial instance prototypes. With the help of our instance pruning approach, MILES could accomplish the further feature mapping and classifier learning more quickly, and thus its efficiency could be significantly improved. Meanwhile, its effectiveness could be maintained or even strengthened sometimes.

The remainder of this paper is organized as follows. In Sect. 2, we review some related work to our research. Our PSIP approach is presented in Sect. 3. In Sect. 4, we evaluate PSIP on two MIL tasks, i.e. drug activity prediction and region-based image categorization. Conclusions are drawn in the last section of the paper.

2 Related Work

Since MIL was first proposed in the context of drug activity prediction, many efforts have been endeavored to address this learning with ambiguous labeling. The first MIL algorithm is called axis-parallel rectangle (APR), which is aimed at finding an APR including at least one instance from each positive bag but excluding all instances from negative bags [1]. A bag is classified as positive if at least one of its instances falls within the APR; otherwise, it is classified as negative. Similarly, Maron and Lozano-Pérez proposed a new concept called diverse density (DD) for MIL, which measures how many different positive bags have instances near a point in the feature space and how far negative instances are from that point [2]. The EM-DD algorithm combines DD with expectation

maximization, aimed at locating the target concept in a more efficient manner [14]. The mi-SVM/MI-SVM algorithm treats the unobservable instance labels as hidden variables and formulates MIL as a mixed integer quadratic program [15]. Ramon and De Raedt extended neural networks to the multiple-instance setting [16]. Wang and Zucker adapted the standard kNN algorithm to the MIL scenario using the Hausdorff distance [17].

Recently, several ISMIL algorithms have been presented, namely DD-SVM [10], MILES [11], MILD [12] and MILIS [13]. The basic idea is mapping each bag into a new feature space called embedding space, which is constructed using some instance prototypes chosen from the training set. Thus, training bags are represented by single feature vectors and the MIL problem is converted to a standard supervised learning one. Then standard SVMs are trained using these bag-level feature vectors. Specifically, DD-SVM uses the DD function [2] to identify instance prototypes. MILES considers all instances in the training set as initial instance prototypes and instance selection is implicitly performed via learning a 1-norm SVM with a linear kernel. MILD performs instance selection based on a conditional probability model. MILIS achieves the initial instance selection by modeling the distribution of the negative population with the Gaussian-kernel-based kernel density estimator. Then it depends on an iterative optimization framework to update instance prototypes and learn a linear SVM.

3 Instance Pruning via Similarity between Paired Instances

In this section, we first describe the details of MILES. The motivation for this work is then provided. Finally, we present our PSIP approach. To describe MILES and PSIP, we need to introduce some notations. Let B represent all training bags and m represent the size of B. We denote the i^{th} bag in B as B_i and the j^{th} instance in that bag as B_{ij}. The bag B_i is composed of n_i instances B_{ij}, $j = 1, 2, \ldots, n_i$.

3.1 MILES

MILES actually follows the work of DD-SVM. Although effective, DD-SVM is very sensitive to labeling noise since the DD value at a point will be exponentially reduced if there is a single instance from a negative bag close to that point. This conclusion has been empirically validated by [11]. To address this issue, MILES presented a more general framework to tackle the MIL problem. The common ground of DD-SVM with MILES lies in the inspiration by the same DD framework [2], while the differences are the usages of DD. As introduced in Sect. 2, DD-SVM uses DD to search for the most likely concepts or instance prototypes, while MILES interprets DD from the feature selection point of view. Specifically, given a concept class C, each concept $t \in C$ is viewed as an attribute or a feature for bags [11]. The value of the feature for bag B_i is defined as

$$h_t(B_i) = \Pr(t|B_i) \ . \tag{1}$$

From the perspective of feature selection, the DD framework appears to be rather restrictive because it always seeks for one and only one feature. Starting from the motivation of attempting to improve the performance by searching for multiple features, MILES further extends the idea of the DD framework in constructing the features. Specifically, MILES regards every instance in the training set as a candidate of target concepts. By interpreting the most-likely-cause estimator [19] as a measure of similarity between a concept and a bag, i.e.

$$\Pr(t|B_i) \propto s(t, B_i) = \max_j \exp(-\gamma \parallel t - B_{ij} \parallel^2) \ , \tag{2}$$

where γ is a scaling factor larger than 0, MILES embeds each bag into a new feature space formed by all instances in the training set. Moreover, MILES indicates that the embedding produces a possibly high-dimensional space when the number of instances in the training set is large. In addition, many features may be redundant or irrelevant because some of the instances might not be responsible for the observed classification of bags, or might be similar to each other. Thus, it is essential and indispensable to select a subset of features that is most relevant to the classification problem of interest. For this purpose, MILES adopts a joint approach that applies a 1-norm SVM to build the classifier and select important features simultaneously. Note that feature selection is essentially instance selection since each feature is defined by an instance.

3.2 Motivation

As indicated by the original authors of MILES, many candidates of target concepts or initial instance prototypes are similar to each other. To solve this problem, they depend on a 1-norm SVM to perform an implicit instance selection. However, instance selection is integrated with classifier learning that is before feature mapping, thus this leads to high complexity for feature mapping and classifier learning since all instances in the training set are used as initial instance prototypes. If some explicit instance pruning is performed before feature mapping, the whole computational complexity will be decreased.

Motivated by the idea of removing similar initial instance prototypes, we try to use the concept of the similarity between paired instances within a bag to conduct instance pruning for MILES. We focus on the structure within a bag, i.e. selecting a pair of instances with the lowest similarity value as initial instance prototypes. In this way, most similar instances are eliminated from the original set of initial instance prototypes used by the original algorithm.

When the above idea comes into mind, one may come up with selecting a number of paired instances with the lowest similarity values from the original set of initial instance prototypes. Nevertheless, this is not a reasonable way and the reason resides in the following fact. First, we do not know the exact pair number. Without doubt, we could determine a principle for this purpose, such as setting a threshold for the exact number. However, this will definitely incur some unnecessary cost. Second, this strategy does not consider bag-level

structure or discriminative information, and may discard small clusters in the feature space where informative features may be located. Finally, as we know, negative instances might be uniformly or randomly distributed in the feature space. Thus, a pair of instances with the lowest similarity in the whole instance feature space may not contain any target concept.

3.3 The Proposed Algorithm

PIPS starts with searching every training bag for a pair of instances with the lowest similarity. For this purpose, we first compute the similarity between any two instances within every training bag. We use the Euclidean distance to evaluate the similarity between all paired instances B_{ij} and B_{ik} in a bag B_i, i.e. $\|B_{ij} - B_{ik}\|$, $i \in \{1, 2, \ldots, m\}$, $j, k \in \{1, 2, \ldots, n_i\}$ and $j \neq k$. Note that the less the distance, the higher the similarity. Then, the paired instances with the lowest similarity within a training bag are used as initial instance prototypes. The pseudo-code for the above discussion has been summarized in Algorithm 1.

Algorithm 1. Pseudo-code for PSIP

Input: Training set B
Output: Set of initial instance prototypes $T = \{\}$

```
for i = 1 to m do
    d_max = −∞
    for j = 1 to n_i − 1 do
        for k = j + 1 to n_i do
            d = ‖B_ij − B_ik‖
            if d > d_max then
                d_max = d, t_1 = B_ij, t_2 = B_ik
    T ⇐ T ∪ {t_1, t_2}
```

4 Experiments and Analysis

4.1 Drug Activity Prediction

The MUSK datasets, MUSK1 and MUSK2, are standard benchmark datasets for MIL [1], which are publicly available from the UCI Machine Learning Repository [20]. These datasets consist of descriptions of molecules and the task is to predict whether a given molecule is active or inactive. Each molecule is viewed as a bag, the instances of which are the different low-energy conformations of the molecule. If one of the conformations of a molecule binds well to the target protein, the molecule is said active; otherwise, the molecule is inactive. MUSK1 contains 47 positive bags and 45 negative bags. MUSK2 contains 39 positive bags and 63 negative bags. MUSK2 shares 72 molecules with MUSK1, but includes more conformations for those shared molecules.

Table 1. Classification accuracies (%) of various algorithms on the MUSK datasets

Algorithm	MUSK1	MUSK2	Mean
$\text{MILES}_{\text{PSIP}}$	88.2 : [87.5, 88.9]	88.8 : [87.9, 89.6]	88.5
MILES [11]	87.4 : [86.1, 88.7]	90.9 : [90.0, 91.8]	89.2
DD-SVM [10]	78.6 : [77.1, 80.1]	85.3 : [84.5, 86.1]	82.0
MILD [12]	82.2 : [80.8, 83.6]	86.2 : [84.5, 87.9]	84.2
MILIS [13]	84.4 : [83.2, 85.6]	88.8 : [87.6, 90.0]	86.6
APR [1]	92.4	89.2	90.8
DD [2]	88.9	82.5	85.7
EM-DD [14]	84.8	84.9	84.9
MI-SVM [15]	77.9	84.3	81.1
mi-SVM [15]	87.4	83.6	85.5

Table 2. Computation time (minutes) of all ISMIL algorithms on the MUSK datasets: time spent on model selection + training time after model selection.

Algorithm	MUSK1	MUSK2
$\text{MILES}_{\text{PSIP}}$	$0.7 + 0.7$	$5.6 + 4.9$
MILES [11]	$1.0 + 1.0$	$180.8 + 218.9$
DD-SVM [10]	$165.4 + 111.5$	$2321.8 + 1170.1$
MILD [12]	$0.5 + 0.4$	$36.6 + 27.4$
MILIS [13]	$2.5 + 2.0$	$705.9 + 814.2$

We used LIBSVM [21] to train all the SVMs for DD-SVM [10], MILES [11], MILD [12] and MILIS [13]. All the parameters required by them were selected according to a twofold cross-validation on the training set. Table 1 reports the means and 95% confidence intervals of the results over ten runs of tenfold cross-validation. For completeness, we have also included the results from some other MIL algorithms. Table 1 shows that APR achieves the best overall performance on the MUSK datasets. However, it is more meaningful to examine the classification accuracy of MILES after using PSIP. The overall classification accuracy of MILES decreases only 0.7% when PSIP is applied. This result indicates that PSIP could guarantee the effectiveness of MILES.

Then we examined the influence of PSIP to MILES with respect to computational efficiency. Table 2 reports the overall computation time of various ISMIL algorithms on the MUSK datasets. The time spent on model selection and the training time after model selection are reported separately. The time spent on model selection is that consumed on selecting the optimal parameter values for every ISMIL algorithm. The training time after model selection is the total training time of ten runs of tenfold cross-validation. All the experiments were conducted on a 3.1 GHz PC. We can find that $\text{MILES}_{\text{PSIP}}$ is superior to other algorithms (except for MILD on MUSK1) in terms of computation time and the efficiency of MILES is obviously improved with the help of PSIP. The speedup for the MUSK2 dataset is more obvious due to the large number of instances in this dataset and the powerful instance pruning ability of PSIP.

Table 3. Classification accuracies (%) of all ISMIL algorithms on the COREL datasets

Algorithm	COREL_{10}	COREL_{20}	Mean
$\text{MILES}_{\text{PSIP}}$	81.6 : [80.7, 82.6]	72.1 : [70.8, 73.4]	76.9
MILES [11]	82.3 : [81.6, 83.0]	71.7 : [70.8, 72.6]	77.0
DD-SVM [10]	80.2 : [79.5, 80.9]	67.8 : [66.7, 68.9]	74.0
MILD [12]	76.2 : [75.3, 77.1]	68.4 : [67.4, 69.4]	72.3
MILIS [13]	82.4 : [81.8, 83.0]	69.6 : [68.7, 70.5]	76.0

4.2 Region-Based Image Categorization

The COREL dataset has been widely used for region-based image categorization. The dataset contains 20 thematically diverse image categories with 100 images of size 384×256 or 256×384 in each category. Each image is segmented into several local regions and features are extracted from each region. The dataset and extracted features are available at `http://www.cs.olemiss.edu/~ychen/ddsvm.html`. Details of segmentation and feature extraction are beyond the scope of this paper and interested readers are referred to [10,11] for further information. Since this is a multiclass classification problem, we apply the one-against-the-rest approach to train 20 binary SVMs. A test bag is assigned to the category with the largest decision value given by the SVMs.

We have conducted two tests for the 10-category and 20-category categorizations. The first 10 categories in the COREL dataset were used for training and testing in the first test while all 20 categories were used in the second test. For each category, we randomly selected half of images as training bags and the remaining half as test bags. Training and testing were repeated for five different random partitions. We used the same experimental setting as in Sect. 4.1 to determine all the parameters needed by various ISMIL algorithms. The average classification accuracies over five different random test sets and the corresponding 95% confidence intervals are provided in Table 3. Overall, the performance of $\text{MILES}_{\text{PSIP}}$ is almost the same as that of MILES and better than that of others.

5 Conclusions

In this paper, we have proposed a novel instance pruning approach to MILES, PSIP, which is based on the similarity between paired instances within a bag. The goal of PIPS is to improve the efficiency of MILES with its effectiveness guaranteed. We have applied PSIP to MILES and examined the influence of PSIP to it from the perspectives of effectiveness and efficiency. The empirical studies on the tasks of drug activity prediction and region-based image categorization demonstrate that our PSIP approach could remarkably improve the computational efficiency of MILES while maintaining or even improving its generalization accuracy, especially for large-scale datasets.

References

1. Dietterich, T., Lathrop, R., Lozano-Pérez, T.: Solving the Multiple Instance Problem with Axis-Parallel Rectangles. Artif. Intell. 89(1-2), 31–71 (1997)
2. Maron, O., Lozano-Pérez, T.: A Framework for Multiple-Instance Learning. In: NIPS, pp. 570–576. MIT Press (1998)
3. Maron, O., Ratan, A.: Multiple-Instance Learning for Natural Scene Classification. In: ICML, pp. 341–349. Morgan Kaufmann (1998)
4. Raykar, V., Krishnapuram, B., Bi, J., Dundar, M., Rao, R.: Bayesian Multiple Instance Learning: Automatic Feature Selection and Inductive Transfer. In: ICML, pp. 808–815. Morgan Kaufmann (2008)
5. Rahmani, R., Goldman, S., Zhang, H., Krettek, J., Fritts, J.: Localized Content-Based Image Retrieval. IEEE Trans. Pattern Anal. Mach. Intell. 30(11), 1902–1912 (2008)
6. Viola, P., Platt, J., Zhang, C.: Multiple Instance Boosting for Object Detection. In: NIPS, pp. 1417–1424. MIT Press (2006)
7. Babenko, B., Verma, N., Dollár, P., Belongie, S.: Multiple Instance Learning with Manifold Bags. In: ICML, pp. 81–88. Morgan Kaufmann (2011)
8. Li, M., Kwok, J., Lu, B.: Online Multiple Instance Learning With no Regret. In: CVPR, pp. 1395–1401. IEEE (2010)
9. Babenko, B., Yang, M., Belongie, S.: Robust Object Tracking with Online Multiple Instance Learning. IEEE Trans. Pattern Anal. Mach. Intell. 33(8), 1619–1632 (2011)
10. Chen, Y., Wang, J.: Image Categorization by Learning and Reasoning with Regions. J. Mach. Learn. Res. 5, 913–939 (2004)
11. Chen, Y., Bi, J., Wang, J.: MILES: Multiple-Instance Learning via Embedded Instance Selection. IEEE Trans. Pattern Anal. Mach. Intell. 28(12), 1931–1947 (2006)
12. Li, W., Yeung, D.: MILD: Multiple-Instance Learning via Disambiguation. IEEE Trans. Knowl. Data Eng. 22(1), 76–89 (2010)
13. Fu, Z., Robles-Kelly, A., Zhou, J.: MILIS: Multiple Instance Learning with Instance Selection. IEEE Trans. Pattern Anal. Mach. Intell. 33(5), 958–977 (2011)
14. Zhang, Q., Goldman, S.: EM-DD: An Improved Multiple-Instance Learning Technique. In: NIPS, pp. 1073–1080. MIT Press (2001)
15. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support Vector Machines for Multiple-Instance Learning. In: NIPS, pp. 561–568. MIT Press (2003)
16. Ramon, J., De Raedt, L.: Multi Instance Neural Networks. In: ICML Workshop on Attribute-Value and Relational Learning (2000)
17. Wang, J., Zucker, J.: Solving the Multiple-Instance Problem: A Lazy Learning Approach. In: ICML, pp. 1119–1126. Morgan Kaufmann (2000)
18. Zhang, Q., Goldman, S., Yu, W., Fritts, J.: Content-Based Image Retrieval Using Multiple-Instance Learning. In: ICML, pp. 682–689. Morgan Kaufmann (2002)
19. Maron, O.: Learning from ambiguity. Ph.D. thesis. Massachusetts Institute of Technology (1998)
20. Blake, C., Merz, C.: UCI Repository of Machine Learning Databases (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
21. Chang, C., Lin, C.: LIBSVM: A Library for Support Vector Machines. Software (2012), `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Interactive Hybrid Systems for Monitoring and Optimization of Micro- and Nano-machining Processes

Dariusz Lipinski and Maciej Majewski

Koszalin University of Technology, Department of Mechanical Engineering
Raclawicka 15-17, 75-620 Koszalin, Poland
{dariusz.lipinski,maciej.majewski}@tu.koszalin.pl
http://kmp.wm.tu.koszalin.pl

Abstract. The article describes a new concept of interactive hybrid systems for monitoring and optimization of micro- and nano-machining processes, which are equipped with voice and visual communication between the human operator and the system. These remote systems contain a speech interface and artificial intelligence. They are presented in exemplary application in the precision grinding process. The developed concept proposes an architecture of the systems equipped with a data analysis layer, process supervision layer, decision layer, communication subsystem by speech and natural language, and visual communication subsystem using voice descriptions. In the system, computational intelligence methods allow for real-time data analysis of monitored processes, configuration of the system, process supervision and optimization based on the process features and quality models. The concept allows for the development of universal and elastic systems which are independent of a type of manufacturing process, machining parameters and conditions.

Keywords: interactive hybrid system, neural networks, intelligent supervision system, interaction between human operators and systems, intelligent interface, voice and visual communication, monitoring and optimization of micro- and nano-machining processes, process quality, measurement data analysis, modern machining process, artificial intelligence.

1 Introduction

In the industry processes of micro- and nano-machining can be performed using a hybrid system for monitoring, optimization and forecasting of the machining process quality, equipped with artificial intelligence methods and a layer of remote voice and visual communication between the system and human operators. This system is presented in exemplary application in the precision grinding processes. It features the possibility for many other applications, future development and experiments. Its main tasks include: modeling of the manufacturing process, assessment of inaccuracy effects, identification of inaccuracy causes, optimization of the process conditions and parameters.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 360–367, 2013.

The scientific aim of the research is to develop fundamentals of building interactive hybrid systems (fig. 1) for monitoring and optimization of micro- and nano-machining processes. The design and implementation of these systems is an important field of research. This concept proposes a novel approach to these systems, with particular emphasis on their ability to be truly flexible, adaptive, human error-tolerant, and supportive both of human operators and intelligent agents in distributed systems architectures. The interactive hybrid system allows

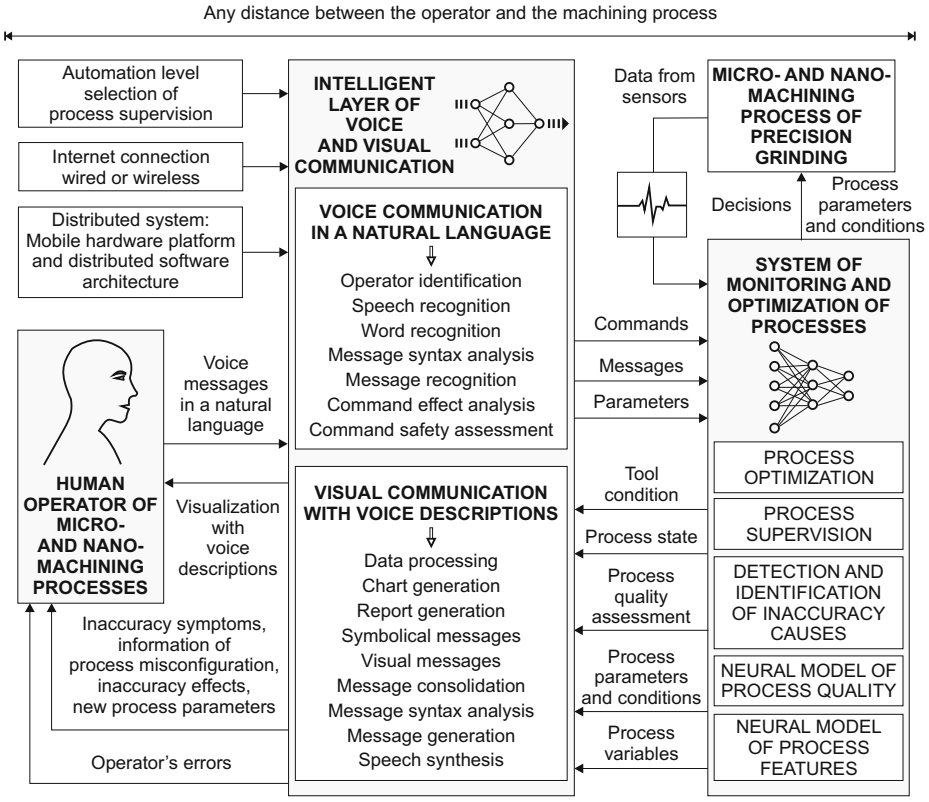


Fig. 1. Concept of interactive hybrid systems for monitoring and optimization of micro- and nano-machining processes

for higher organization level of manufacturing processes, which is significant for their efficiency and humanization. Decision and optimization systems can be remote elements of manufacturing processes. The design of the proposed system can be considered as an attempt to create a standard interactive system for monitoring and optimization of machining processes. It is very significant for the development of new effective and flexible manufacturing methods.

2 The State of the Art

There is a need for remote systems of monitoring and optimization of machining processes in reconfigurable manufacturing systems to reduce bottlenecks that occur in associated tasks to be performed by these systems using technological devices. The tasks include: modeling of the process features and quality, assessment of inaccuracy effects, identification of inaccuracy causes, optimization of the process conditions and parameters. These bottlenecks can occur as a result of the mass production of custom products.

The sustainability of existing manufacturing resources and enhancement of the machining efficiency is an important field of applied research. The current research and recent advances in development of prototypes of systems for monitoring and optimization of machining processes are described in articles [1,2]. Those systems consist of computational algorithms for sensor base monitoring and control, and simulations of virtual machining processes. Those systems have been accepted for real-time decision making, sustainable development and efficient use of machining resources. In many potential applications of these monitoring and optimization systems, the limiting factor may be an ability of the system to model the process features and quality, process the inaccuracy symptoms, solve the misconfiguration issues, compensate the inaccuracy effects, determine the process conditions and parameters.

This article offers an approach by using the developed concept of the interactive hybrid system of monitoring and optimization of the processes of micro- and nano-machining to deal with the above problems. Selected article [2] presents innovative solutions in supervision of precise grinding processes and development of a hybrid system for monitoring, optimization and forecasting of machining process quality. Articles [3,4,5,6,7,8,9] describe the developed solutions in intelligent voice communication between human operators and technical devices.

3 Description of the System

The developed concept proposes an architecture of the interactive hybrid system for monitoring and optimization, which is equipped with a data analysis layer, process supervision layer, decision layer, communication subsystem by speech and natural language, and visual communication subsystem using voice descriptions. The structure of the system is presented in abbreviated form on Fig. 2. The numbers in the cycle represent the successive phases of information processing. The novelty of the system (fig. 3) consists of inclusion of adaptive intelligent layers for data analysis, supervision and decision. The system is also capable of analysis of the supervised machining process, configuration of the supervision system, neural modeling of process features, neural modeling of process quality, detection of the inaccuracies, estimation of the inaccuracy results, compensation of the inaccuracy results, and selection of the machining parameters and conditions. The core of the system consists of the following process models: the neural

model of the optimal process parameters for determination of optimal values of the process features, and the neural model for assessment of influence of the measured process features on the process quality parameters.

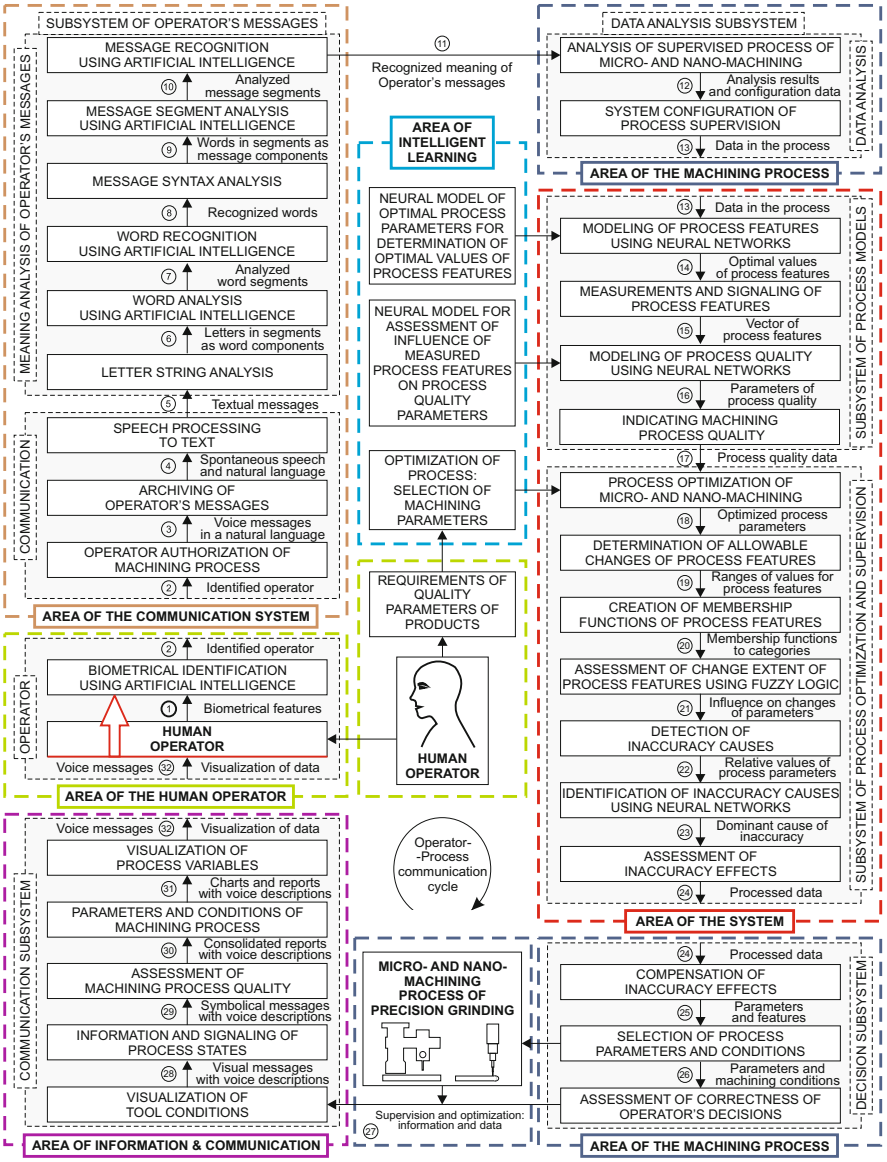


Fig. 2. Implementation structure of hybrid systems for monitoring and optimization of micro- and nano-machining processes using voice and visual communication

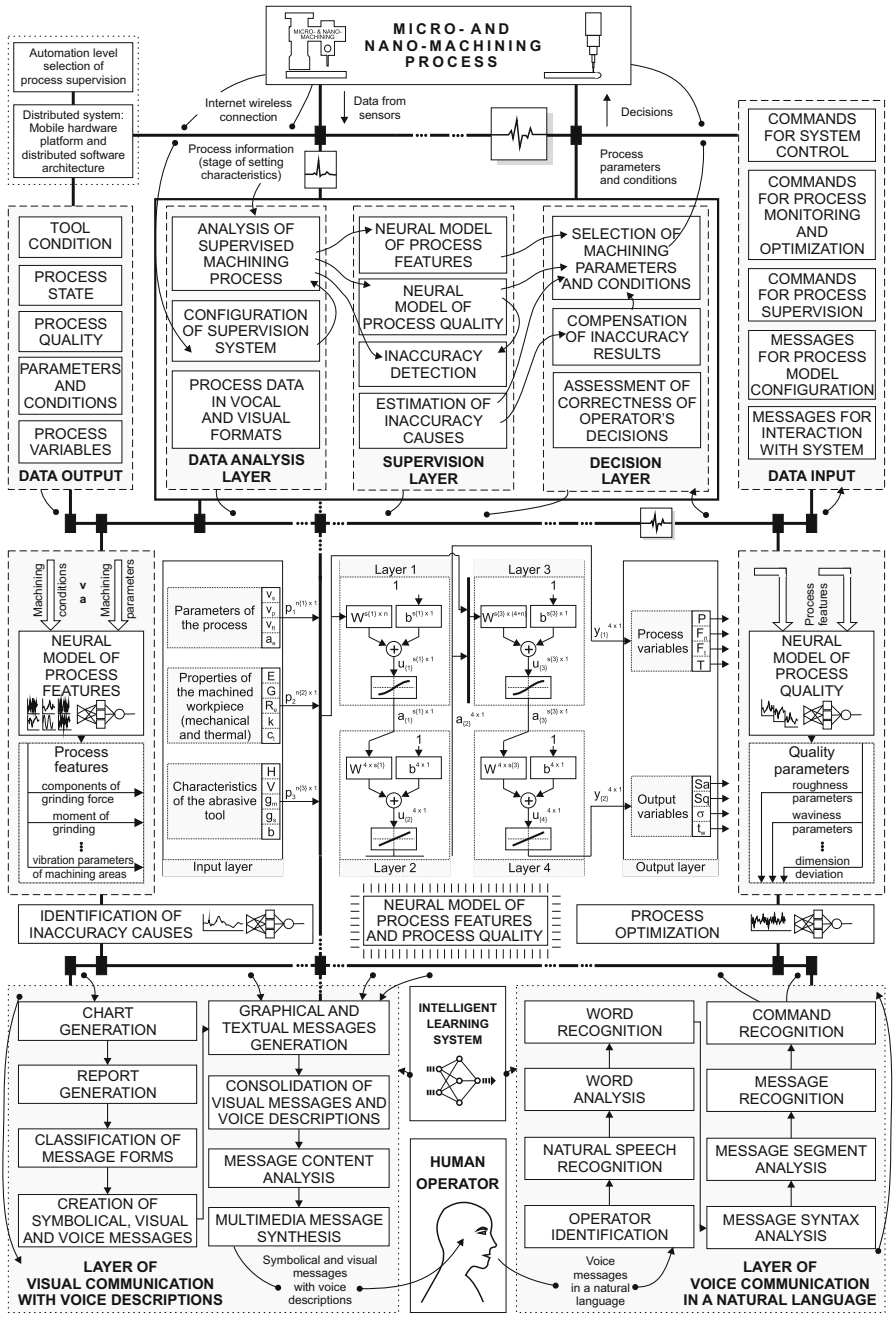


Fig. 3. Architecture of interactive hybrid systems for monitoring and optimization of micro- and nano-machining processes with the core of neural models of the process features and quality

The neural models of the process features and quality are used in a subsystem for detection of inaccuracies and optimization of machining parameters. The system also consists of mechanisms (fig. 4) for meaning analysis of operator's messages and commands given by voice in a natural language, and various visual communication forms with the operator using voice descriptions.

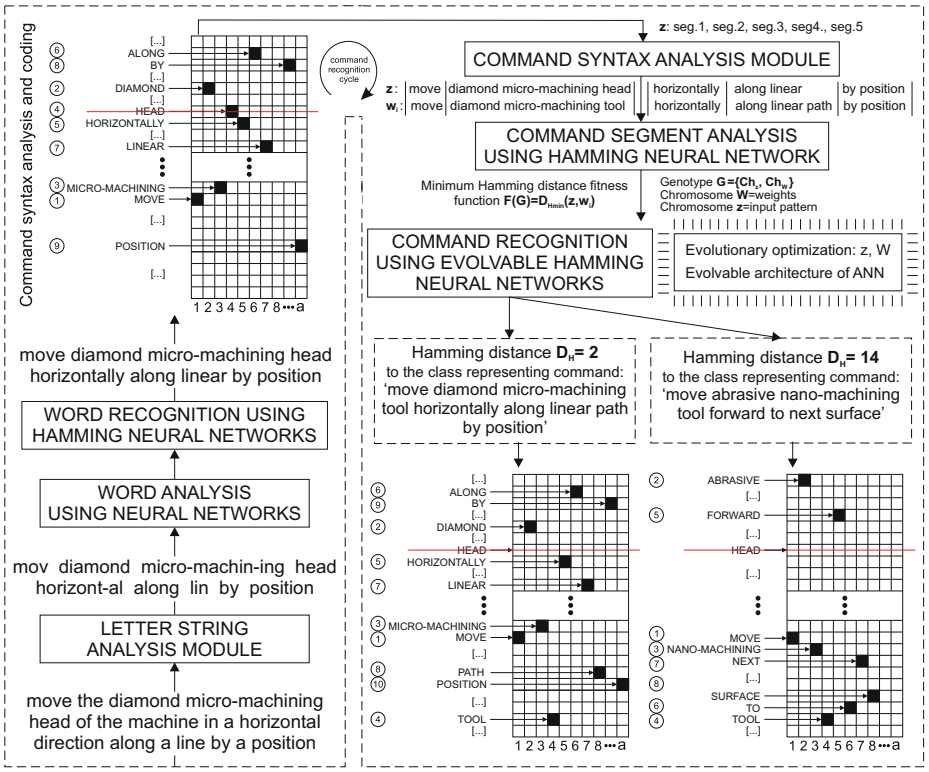


Fig. 4. Illustration of a cycle of meaning analysis of the human operator's messages and commands given by voice in a natural language using evolvable neural networks

The interaction between the operator and the system by speech and natural language contains intelligent mechanisms for operator biometric identification, speech recognition, word recognition, recognition of messages and commands (fig. 5), syntax analysis of messages, and safety assessment of commands. The interaction between the system and the operator using visual messages with voice descriptions includes intelligent mechanisms for generation of graphical and textual reports, classification of message forms, generation of messages in the graphical and textual forms, consolidation and analysis of message contents, synthesis of multimedia messages.

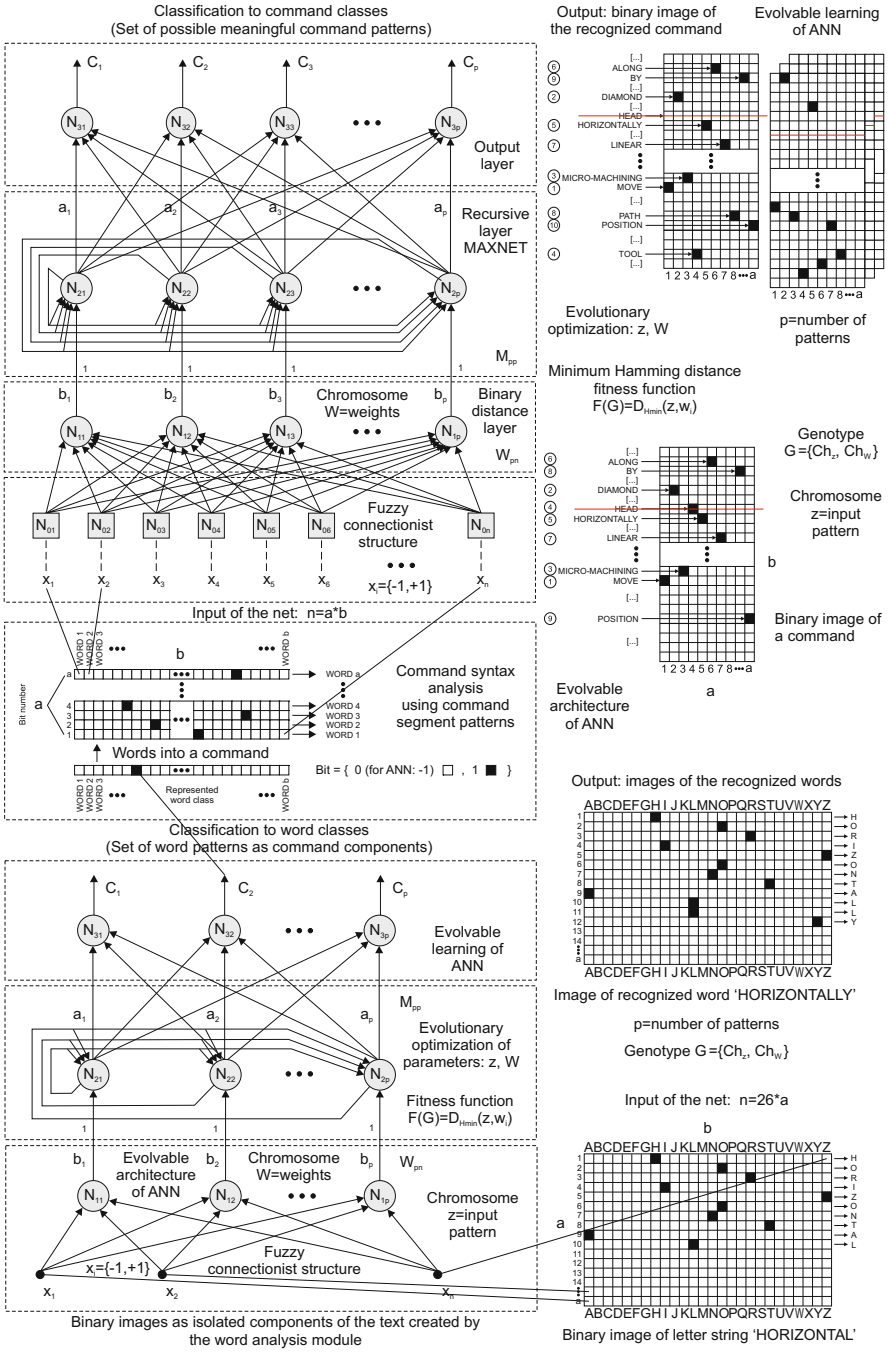


Fig. 5. Evolvable fuzzy neural networks for word and command recognition

4 Conclusions and Perspectives

The proposed concept of the interactive hybrid systems for monitoring and optimization of machining processes, equipped with a speech interface and artificial intelligence, allows for the development of universal and elastic systems which are independent of a type of manufacturing process, machining parameters and conditions. The condition of effectiveness of the system is to equip it with intelligent mechanisms for modeling of the process features and quality, assessment of inaccuracy effects, identification of inaccuracy causes, optimization of the process conditions and parameters. The experimental results of the proposed system show its promising performance. The concept can be used for further development and experiments. The system is both effective and flexible which makes its applications possible. It features the universality of application of the developed artificial intelligence algorithms. The hybrid system allows for more robustness to human's errors. The proposed complex solution also eliminates scarcities of the typical co-operation between human operators and technological devices.

Acknowledgements. The authors would like to thank The National Science Center in Poland (NCN) who provided financial support for this work.

References

1. Groover, M.P.: Fundamentals of Modern Manufacturing. Wiley & Sons, Inc. (2007)
2. Lipiński, D., Kacalak, W.: Assessment of the Accuracy of the Process of Ceramics Grinding with the Use of Fuzzy Interference. In: Beliczynski, B., Dzielinski, A., Iwanowski, M., Ribeiro, B. (eds.) ICANNGA 2007. LNCS, vol. 4431, pp. 596–603. Springer, Heidelberg (2007)
3. Kacalak, W., Majewski, M.: New Intelligent Interactive Automated Systems for Design of Machine Elements and Assemblies. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part IV. LNCS, vol. 7666, pp. 115–122. Springer, Heidelberg (2012)
4. Kacalak, W., Majewski, M.: Effective Handwriting Recognition System Using Geometrical Character Analysis Algorithms. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part IV. LNCS, vol. 7666, pp. 248–255. Springer, Heidelberg (2012)
5. Kacalak, W., Majewski, M.: Natural Language Human-Robot Interface Using Evolvable Fuzzy Neural Networks for Mobile Technology. In: Huang, D.-S., Jo, K.-H., Lee, H.-H., Kang, H.-J., Bevilacqua, V. (eds.) ICIC 2009. LNCS, vol. 5754, pp. 480–489. Springer, Heidelberg (2009)
6. Majewski, M., Zurada, J.M.: Sentence Recognition Using Artificial Neural Networks. Knowledge-Based Systems 21(7), 629–635 (2008)
7. Majewski, M., Kacalak, W.: Intelligent System for Natural Language Processing. In: Huang, D.-S., Li, K., Irwin, G.W. (eds.) ICIC 2006. LNCS (LNAI), vol. 4114, pp. 742–747. Springer, Heidelberg (2006)
8. Stuart, K.D., Majewski, M., Trelis, A.B.: Intelligent Semantic-Based System for Corpus Analysis through Hybrid Probabilistic Neural Networks. In: Liu, D., Zhang, H., Polycarpou, M., Alippi, C., He, H. (eds.) ISNN 2011, Part I. LNCS, vol. 6675, pp. 83–92. Springer, Heidelberg (2011)
9. Stuart, K.D., Majewski, M.: Evolvable Neuro-fuzzy System for Artificial Creativity in Linguistics. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 46–53. Springer, Heidelberg (2008)

Deep Neural Networks for Source Code Author Identification

Upul Bandara and Gamini Wijayarathna

Department of Industrial Management, University of Kelaniya, Sri Lanka
upulbandara@gmail.com, gamini@kln.ac.lk

Abstract. Plagiarism and copyright infringement are major problems in academic and corporate environments. Importance of source code authorship attribution arises as it is the starting point of detection for plagiarism, copyright infringement and law suit prosecution etc. There have been many research regard to this topic. Majority of these researches are based on various algorithms which compute similarity amongst source code files. However, for this Paper we have proposed Deep Neural Network (DNN) based technique to be used for source code authorship attribution. Results proved that DNN based author identification brings promising results once compared the accuracy against previously published research.

Keywords: Restricted Boltzmann Machine, Deep Neural Networks, Source Code Authorship Attribution.

1 Introduction

Source code authorship attribution has many applications in different fields such as source code plagiarism detection, digital forensics, and intellectual property infringement [1]. Since magnitude of source code repositories are growing very rapidly, it is impractical to use manual techniques for source code authorship attribution. Therefore, automatic techniques could be ideal solutions for identifying authors of source codes.

In this paper we investigate deep neural networks for source code authorship attribution task. Training deep neural networks are known to be hard. It is empirically shown that the standard backpropagation algorithm could easily get stuck in a local minimum. Therefore, until recently neural networks have been limited to one or two hidden layers [2]. However, training deep neural networks using greedy layer-wise pre-training before fine-tuning using backpropagation enables to overcome above limitation [3]. Therefore, we employed that technique to train our deep neural network. Our system was evaluated with several datasets and results have shown that performance is very close to the state of art techniques in the source code identification field.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 368–375, 2013.

2 Previous Work

A few research papers have been written on source author identification using machine learning techniques. Following briefs such important techniques we found are useful during literature surveying phase.

Lange and Mancoridis [4] have proposed a source code authorship attribution method using source code metric histograms and genetic algorithm. From source code metrics, an optimum set of source code metrics were selected using genetic algorithms. Selected metrics were used as the input for the nearest neighbor classifier and the system is capable of identifying the true author of each source code file with 55 percent accuracy.

Burrows and Tahaghoghi [5] described a system for source code author identification using an information retrieval approach. First, the n-gram tokens were generated from each source code file. Then the generated tokens were indexed in a search engine. During the testing period each test document was converted into a collection of n-gram tokens and was compared with indexed source code files. According to the paper, their system is capable of identifying true authors with 67 percent of accuracy.

Frantzeskou et al. [1] described a technique called, Source Code Author Profiles (SCAP) for authorship attribution. SCAP is based on generating byte level n-gram author profiles. In order to classify a test source code file, its profile is compared with the pre-calculated training author profiles and the most likely author is the one who has the least dissimilar profile. According the paper, accuracy of the SCAP method is very equal to 100 percent.

Application of decision tree techniques and programming style metrics for source code author identification are described by Elenbogen and Seliya [6]. Performance of their system was evaluated with a dataset consisted of 82 source code files belonging to 12 authors and they have achieved 74.70 percent of accuracy. Shevertalov et al. [7] described a source code discretization based method for generating source code author profiles. They evaluated their system with a dataset consisting of 75 000 Java source code files, belonging to 20 authors and reported 75 percent of classification accuracy.

3 Training the Deep Neural Network

3.1 Restricted Boltzmann Machine (RBM)

RBM is a bipartite graph in which visible units represent the input data and hidden units represent features using undirected connections [8]. RBM has been used for various supervised and unsupervised applications such as, dimensionality reduction, classification, collaborative filtering, and clustering. RBM was invented by Smolensky [9], however, it became very popular after introducing fast learning algorithms by Hinton [10]. RBM is an energy based model and the joint energy between visible and hidden units is defined by Eq. 1.

$$E(\boldsymbol{v}, \boldsymbol{h}; \theta) = -\sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij}\, v_i h_j - \sum_{i=1}^{D} b_i v_j - \sum_{j=1}^{F} a_j h_j \tag{1}$$

Where $\theta = \{\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{b}\}$ represent the parameters of the model. W_{ij} represents the symmetric weight between i^{th} visible unit and the j^{th} hidden unit. b_i denotes connection between bias term and the i^{th} visible unit. a_j denotes the connection between bias unit and the j^{th} hidden unit. $\boldsymbol{v}$ and $\boldsymbol{h}$ represent the visible and hidden vectors respectively.

The joint probability distribution of visible and hidden units is given by Eq. 2 and Eq. 3 and $Z(\theta)$ is known as the partition function.

$$P(\boldsymbol{v}, \boldsymbol{h}; \theta) = \frac{1}{Z(\theta)} e^{-E(\boldsymbol{v}, \boldsymbol{h}; \theta)} \tag{2}$$

$$Z(\theta) = \sum_{v} \sum_{h} e^{-E(\boldsymbol{v}, \boldsymbol{h}; \theta)} \tag{3}$$

The probability that the model assigns to the visible vector v is given by Eq. 4.

$$P(\boldsymbol{v}; \theta) = \frac{1}{Z(\theta)} \sum_{h} e^{-E(\boldsymbol{v}, \boldsymbol{h}; \theta)} \tag{4}$$

By taking the derivatives of Eq. 1 with respect to model parameters we can derive following learning rules.

$$\frac{\partial log P(\boldsymbol{v}; \theta)}{\partial W_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{\text{model}} \tag{5}$$

$$\frac{\partial log P(\boldsymbol{v}; \theta)}{\partial b_{ij}} = \langle v_i \rangle_{data} - \langle v_i \rangle_{model} \tag{6}$$

$$\frac{\partial log P(\boldsymbol{v}; \theta)}{\partial a_{ij}} = \langle h_i \rangle_{data} - \langle h_j \rangle_{model} \tag{7}$$

Where, $\langle . \rangle_{data}$ is the expected value evaluated on data distribution and $\langle . \rangle_{model}$ shows the expected value calculated on model distribution.

Contrastive divergence is a recipe widely used for training RBM. It approximates the gradient of log likelihood of RBM using Gibbs sampling. For this research we used contrastive divergence for training RBMs.

4 Source Code Authorship Attribution System

Under this heading we describe the main components of our source code authorship attribution system.

Fig.1 (a) shows the high-level architecture of the system. Source codes were converted to code metrics and used as the input for the system. Although, a large number of metrics can be generated from source code files, Lange and Mancoridis [4] have conducted an extensive research and identified eight source code metrics, which shows the best performance in the context of source code author identification. However, some of

these metrics are not fully independent from others. For example, trail-space, and trail-tab (measure the trailing whitespaces and tabs at the end of a line) are very similar to each other. Therefore, we represent these two as a single metric called "TrailTabSpace-Calculator" in our system. Similarly, we combined all the metrics which are not fully independent from others, as single metrics. Moreover, we have introduced three more code metrics. Altogether, there were nine metrics as shown in the Table 1.

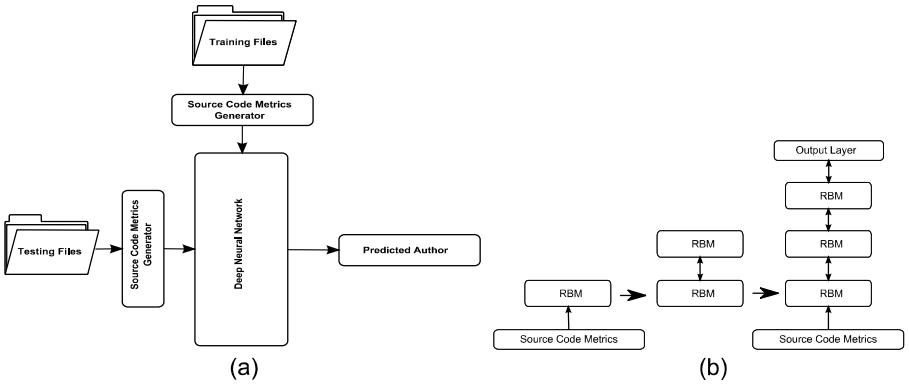


Fig. 1. (a) The high-level architecture of the source code author attribution system, (b) The steps required to implement the deep neural network with three hidden layers and one output layer

Table 1. Source code metrics used for source code author identification

Metric name	Code	Description
LineLengthCalculator	LLC	This metric measures the number of characters in one source code line
LineWordsCalculator	LWC	This metric measures the number of words in one source code line
AccessCalculator	ACL	Java uses the four levels of access control: public, protected, default and private. This metric calculates the relative frequency of these access levels used by the programmers
CommentsFrequency-Calculator	CFC	Java uses three types of comments. This metric calculates the relative frequency of those comment types used by the programmers
IndentifiersLengthCalculator	ILC	This metric calculates the length of each identifier of Java programs
InLineSpaceInlineTab-Calculator	INT	This metric calculates the whitespaces that occurs on the interior areas of non-whitespace lines
TrailTabSpaceCalculator	TTS	This metric measures the whitespace and tab occurrence at the end of each non-whitespace line
UnderscoresCalculator	USC	This metric measures the number of underscore characters used in identifiers
IndentSpaceTabCalculator	IST	This metric calculates the indentation whitespaces used at the beginning of each non-whitespace line

The source code metrics as it is cannot be used as the input of the source code author identification system. Therefore, metrics were converted to a stream of tokens. This process is better explained with an example.

Table 2. Source code metric and token frequencies generated by "LineLengthCalculator" metric

Line length	Number of occurrences	Token	Token frequency
1	2	LLC_1	2
8	12	LLC_8	12
14	3	LLC_14	3
21	1	LC_21	1

Consider a metric generated by the "LineLengthCalculator" for a particular source code file as shown in the first two columns of the Table 2. This code metric generates a set of tokens and token frequencies as depicted in last two columns of the Table 2. Similarly, we converted all the metrics generated from source codes into a set of tokens with token frequencies. These tokens together with token frequencies are used as the input for our system.

Fig.1 (b) shows the sequence of operations required to construct the deep neural network of the system. Source code metrics generated from the "Source Code Metrics Generator" module of the system are used as the input for the first RBM. Output of the fist RBM is used as the input of the second RBM. Following the same procedure we trained three RBMs. Finally, the pre-trained DNN was created by adding the output layer on top of the last RBM.

5 Training and Evaluation

We have measured the performance of our system using five datasets. Find below the details about these five datasets.

Dataset I was created by LangeandMancoridis [4]. It consists of Java source code files belonging to 10 authors. These files were extracted from the Sourcefore[1] website. Then we created Dataset II by downloading Java source code files from free and open source projects in the Internet. It consists of Java source code files belonging to 10 authors. Dataset III consists of Java source code files belonging to eight authors and it was created by using source codes shipped with the Java Development Kit[2]. Dataset IV consists of Java files downloaded from Plant Source Code[3] website and it contains code files belonging to five authors. Finally, we created Dataset V by using Java codes freely available with several programming books published by ApressInc[4]. Details information about these five datasets is given in Table 3.

[1] `http://sourceforge.net/` [Accessed: 2012, March 12]
[2] `http://www.java.com/en/download/index.jsp` [Accessed: 2012, April 02]
[3] `http://www.planet-source-code.com/` [Accessed: 2012, August 01]
[4] `http://www.apress.com/` [Accessed: 2012, August 03]

Table 3. Detailed information for datasets used in this study. Program lengths are measured by means of Lines of Codes (LOC)

Property	I	II	III	IV	V
Number of authors	10	10	8	5	9
No of source code files	1644	780	475	131	520
Minimum files per author	61	28	33	20	5
Maximum files per author	377	128	118	36	118
Size of the smallest file (LOC)	7	28	91	8	20
Size of the largest file (LOC)	3691	15052	4880	654	1135
Average LOC per author	29857	44620	26690	2452	6268

We divided first dataset into three subsets called as training, cross validation, and testing. The rest of the datasets were divided into two training and testing subsets. The system also consists of several hyper-parameters as given below and which were estimated using the cross validation dataset.

1. Learning rate of RBMs.
2. Regularization parameter of the neural network.
3. Number of logistic units in hidden layer 1, hidden layer 2, and hidden layer 3.

Presently, manual search and grid search are the most widely used techniques for estimating optimum values for the hyper-parameters of learning systems. However, Bergstra and Bengio [11] have shown that random search is more effective than manual and grid search. Hence, we employed random search as the method for finding optimum values for the hyper-papers of our system.

We initiated the training process with a batch of 50 source code files. With each iteration training dataset size was increased by 50 and process continued until 850 files in the final training dataset. For each iteration, we tested the accuracy of the system, 25 times with randomly selected values for hyper-parameters using the cross validation dataset. The accuracy of the cross validation dataset was plotted as scatter graphs as shown in Fig 2(a) to Fig 2(q). Next, we selected the best cross validation accuracy from each iteration and drew the cross validation accuracy vs. training batch size graph as depicted in Fig 2(s). We used it has a tool for selecting training batch sizes and hyper-parameter values for other datasets.

Table 4. Percentage accuracies on five datasets mentioned in Table 3

System	I	II	III	IV	V
Lange and Mancoridis[12]	55.00%	-	-	-	-
Bandara and Wijayarathna[13]	86.64%	-	-	-	-
Bandara and Wijayarathna[14]	92.82%	93.64%	90.78%	77.42%	89.62%
This paper	93.65%	93.22%	93.62%	78.12%	89.62%

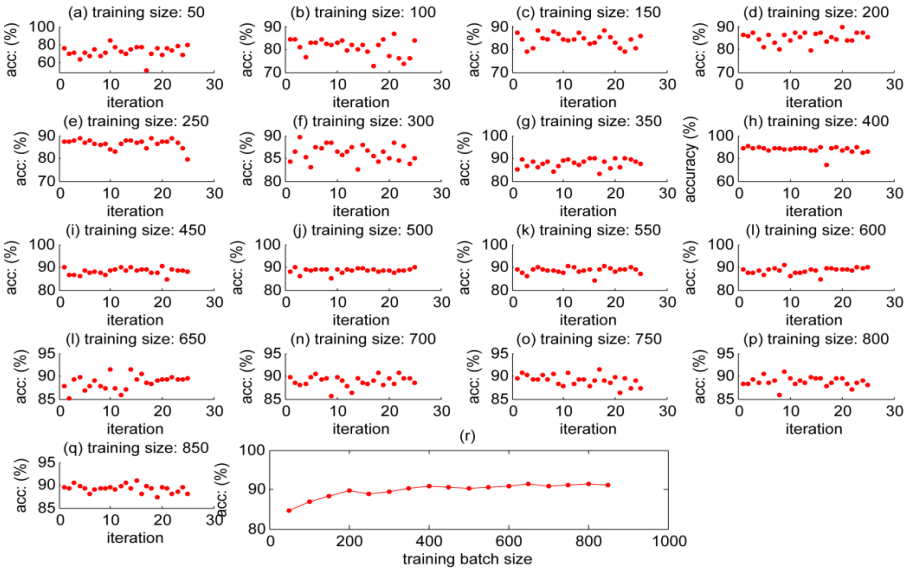


Fig. 2. Classification accuracy of cross validation dataset as a function of hyper-parameters of the system. (a)- (q) cross validation accuracy vs. number of iterations for different training batch sizes (r) cross validation accuracy vs. training batch size

6 Conclusion and Future Works

The paper has investigated Deep Neural Networks for source code authorship attribution. There we have used Restricted Boltzmann Machine for pre-training hidden layers of deep neural networks. Test results confirm a considerable improvement in accuracy compared to the existing published methods for author identification.

Literature also suggest that, for pre-training of deep neural network, not only the Restricted Boltzmann Machine algorithm but also other algorithms like Autoassociators, Denoising Auto Encoders, and Predictive Sparse Coding are used. Therefore, it should be an interesting alternative to investigate performance of other pre-training algorithms in this regard.

References

1. Frantzeskou, G., Stamatatos, E., Gritzalis, E., Katsikas, S.: Source Code Author Identification Based on N-gram Author Profiles. In: Maglogiannis, I., Karpouzis, K., Bramer, M. (eds.) Artificial Intelligence Applications and Innovations. IFIP, vol. 204, pp. 508–515. Springer, Boston (2006)
2. Larochelle, H., Bengio, Y., Louradour, J., Lamblin, P.: Exploring Strategies for Training Deep Neural Networks. Journal of Machine Learning Research 10, 1–40 (2009)
3. Hinton, G.H.: Reducing the dimensionality of data with neural networks. Science, 504–507 (2006)

4. Lange, R., Spiros, M.: Using code metric histograms and genetic algorithms to perform author identification for software forensics. In: 9th Annual Conference on Genetic and Evolutionary Computation, London, pp. 2082–2089 (2007)
5. Burrows, S., Tahaghoghi, S.: Source Code Authorship Attribution using N-Grams. In: Wu, A. (ed.) Source Code Authorship Attribution using N-Grams, Melbourne, Melbou Australiarne, pp. 32–39 (2007)
6. Elenbogen, B., Seliya, N.: Detecting outsourced student programming assignments, pp. 50–57 (January 2008)
7. Shevertalov, M., Kothari, J., Stehle, E., Mancoridis, S.: On the Use of Discretized Source Code Metrics for Author Identification. In: Proceedings of the 2009 1st International Symposium on Search Based Software Engineering, Washington, DC, USA, pp. 69–78 (2009)
8. Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.-R., Jaitly, N., Senior, A., Vanhoucke, V.: Deep Neural Networks for Acoustic Modeling in Speech Recognition. IEEE Signal Processing Magazine, 82–97 (November 2012)
9. Smolensky, P.: Information processing in dynamical systems: Foundations of harmony theory. In: Parallel Distributed Processing Explorations in the Microstructure of Cognition, pp. 194–281 (1986)
10. Hinton, G.: To recognize shapes, first learn to generate images. Progress in Brain Research 165(3), 535–547 (2007)
11. Bergstra, J., Bengio, Y.: Random Search for Hyper-Parameter Optimization. The Journal of Machine Learning Research, 281–305 (2012)
12. Lange, R., Mancoridis, S.: Using code metric histograms and genetic algorithms to perform author identification for software forensics. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, GECCO 2007 (2007)
13. Bandara, U., Wijayarathna, G.: A machine learning based tool for source code plagiarism detection. International Journal of Machine Learning and Computing 1(4), 337–343 (2011)
14. Bandara, U., Wijayarathna, G.: Source code author identification with unsupervised feature learning. Pattern Recognition Letters 34(3), 330–334 (2013)

Desert Vehicle Detection Based on Adaptive Visual Attention and Neural Network

Jinjian Zhang and Xiaodong Gu*

Department of Electronic Engineering, Fudan University,
Shanghai 200433, China
{11210720038,xdgu}@fudan.edu.cn

Abstract. This paper proposes a novel desert vehicle detection method using adaptive visual attention model and pulse coupled neural network. Firstly, an adaptive phase spectrum of quaternion Fourier transform (APQFT) model is proposed to generate and weight information channel of background, intensity, and image colors into a visual saliency map. This model has learning ability and weight values are calculated by least square method. Meanwhile, using bilinear transformation improves computing efficiency. Secondly, using pulse coupled neural network (PCNN) detects regions of interests (ROIs) and using scale-invariant feature transform (SIFT) extracts features of ROIs. Finally, using hierarchical discriminant regression (HDR) tree identifies vehicle areas .Experimental results shows that proposed method is faster with higher recognition rate and lower false alarm than Zheng's and Thomas's methods.

Keywords: desert vehicle detection, adaptive visual attention, least square method, pulse coupled neural network, SIFT feature, HDR tree.

1 Introduction

In vehicle rescue field, because of the high temperature, strong sand storms and weak GPS signal in the complex and variable desert environment, vehicles are more prone to break down and difficult to detect. Firstly, the resolution of remote sensing image is very low compared to those shoot by traditional cameras. Secondly, vehicle unit is small in the remote sensing image so that it's hard to extract model and vehicle information which is important to traditional detection algorithms. Thirdly, interfering factors in the background whose areas and colors is similar to these of vehicles, such as plants and hills, regions of interest (ROI) are hard to be located. For these reasons, traditional vehicle detection algorithms have the disadvantage of low recognition rate and high false alarm rate in desert sense, which cannot meet the requirements of high computational efficiency and accuracy in desert vehicle rescue.

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 376–383, 2013.

2 Related Work

In recent years, visual attention is gradually applied to pattern detection field and obtains the desired performance result. Itti [2] and Walther [3] propose the model that imitates human vision mechanism to calculates the salient map which the intensity of each pixel represents interesting extent. After that, models in frequency domain are proposed, such as phase spectrum of quaternion Fourier transforms (PQFT) [4]. This model utilizes quaternion to composite image color, intensity and motion can process multi-channel information simultaneously to achieve better performance.

In our work, desert images usually have large size and background color is also similar. For these characteristics, this paper proposes an adaptive quaternion visual attention model, which adds a background adaptive channel to original model to enhance background color antagonism and also introduce bilinear transformation to improve computational efficiency. Moreover, least square method is used to calculate a specific weight for each channel, which provides visual attention model learning ability. After that, we combine our visual attention model with pulse-coupled neural network (PCNN) [5]. Use the pulse propagation effect in PCNN to extract ROIs in desert images. Afterwards, scale-invariant feature transform (SIFT) [6] features are extracted from each ROI and then to be compared by trained hierarchical discriminate tree (HDR) [7]. Finally, appraisement standard is established to detect vehicle areas.

3 Algorithm Structure

3.1 Adaptive Visual Attention Model

(a) Construct Quaternion Image

Assume the input image at time t as $F(t), t = 1, 2, \cdots, N$ in which N is the number of frames. Each frame has four channels: one motion channel, tow color difference channel and one intensity channel, which are all combined as following equation, $q(t) = M(t) + R_G(t)\mu_1 + B_Y(t)\mu_2 + I(t)\mu_3$, where $q(t)$ is the input image, $\mu_1 \perp \mu_2$, $\mu_2 \perp \mu_3$, $\mu_3 \perp \mu_1$, $\mu_3 = \mu_1\mu_2$. Four channels are obtained by following ways respectively. Firstly, the red, green and blue information in each frame are recorded as $r(t), g(t), b(t)$. The intensity channel I(t) can be further calculated as $I(t) = (r(t) + g(t) + b(t))/3$. The quaternion color model imitates the human vision system. There are many color antagonisms in our brain, namely neurons are excited by some colors whereas inhibited by another one. In human's visual cortex, red/green, green/red, blue/yellow and yellow/blue are four mainly color couples. Therefore, two color channels in quaternion color model can be obtained by RG(t)=R(t)-G(t) and BY(t)=B(t)-Y(t).

(b) Construct Adaptive Background Channel

The motion channel in quaternion image is utilized to calculate image variation. Due to fact that the vehicle detection uses the static image, we set this channel as 0.

Meanwhile, although there are a lot of interfering factors in desert, such as plants, hills, ravines and so on, the overall background color is similar. Thus we add a channel into the model to increase the inhibition effect against background color, $W(t)=\frac{|R(t)-R_b(t)|+|G(t)-G_b(t)|+|B(t)-B_b(t)|}{3}$, where R_b, G_b, B_b is the average level of background's red, green and blue level. To better distinct the target area, proposed algorithm further revises the quaternion image to be 1.5 times the original one. The new one decreases the calculation time and also enlarges the object area. The processed quaternion image is $q(t)$.

(c) Calculate Saliency Map

Ell and Sangwine first propose the quaternion transform of color images, which we can convert the formula using the inverse transform as

$$f_i[n,m]=\frac{1}{\sqrt{MN}}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}e^{\mu_1 2\pi\left(\left(\frac{mv}{M}\right)+\left(\frac{nu}{N}\right)\right)}F_i[u,v] \tag{1}$$

Then the frequency expression $Q(t)$ of quaternion image $q(t)$ can be expressed as $Q(t)=\|Q(t)\|e^{\mu\phi(t)}$, where ϕ(t) is the phase spectrum of Q(t), μ is quaternion unit. We make $\|Q(t)\|=1$ so that $Q(t)$ only includes the phase spectrum information. To provide this model learning ability for specific environment, weight is considered for each channel. Then, $\tilde{q}(t)$ can be expressed as: Where ϕ(t) is the phase spectrum of *Q(t)*, μ is quaternion unit. We make $\|Q(t)\|=1$ so that $Q(t)$ only includes the phase spectrum information. To provide this model learning ability for specific environment, weight is considered for each channel. Then, $\tilde{q}(t)$ can be expressed as,

$$\tilde{q}(t)=w_1 a(t)+w_2 b(t)\mu_1+w_3 c(t)\mu_2+w_4 d(t)\mu_3 \tag{2}$$

where w_1, w_2, w_3, w_4 could be set by least square method using training Samples and $a(t), b(t), c(t), d(t)$ represents four parts of quaternion.

The saliency map is $S_M(t)=G*\tilde{q}(t)^2$. We zoom in $S_M(t)$ to the original size using bilinear interpolation to obtain the final saliency map we need $S_M(t)'$. The G in the above equation is a two-dimensional low-pass Gaussian filter ($\sigma=8$).

3.2 Pulse Coupled Neural Network

Unit-linking PCNN comprises of three parts, which are receptive field, modulation field and pulse generator respectively. Each neuron in the unit-linking PCNN corresponds to one pixel in the saliency map. The intensity I of pixel j inputs to *F* channel of corresponding pixel as $F_j=I_j$. Making seed neurons (pixel that intensity is greater than preset threshold) initially (intensities set to 1), then the pulses produced by these pixels propagate to the 8-neighborhood. *L* is the other input channel of pixels which

takes neighbor pixels' information in. The unit-linking PCNN simplify L to a Step Function, see equation (3). Pixel j in the neighborhood receives pulse and determines whether fire or not based on its corresponding F and L channel as $U_j = F_j\left(1+\beta_j L_j\right)$. Where β is a preset value and pixel set on fire if U_j is greater than the threshold. Then, firing pixels will propagate the pulses to the neighborhood and those which have fired cannot fire again. The above procedure can processes saliency map to detect the region which may contains vehicles.

$$L_j = S\left[\sum_{k\in N(j)} Y_k(t)\right] = \begin{cases} 1, & \sum_{k\in N(j)} Y_k(t) > 0 \\ 0, & else \end{cases} \tag{3}$$

3.3 Scale Invariant Feature Transform

SIFT [6] is the local feature that is fast to calculate and has rotation, scale, translation and intensity invariance. Meanwhile, SIFT features have the advantage of high uniqueness, fast calculation speed and strong scalability.

3.4 Hierarchical Discriminant Regression Tree

Hierarchical discriminant regression tree (HDR) [7] is a memory tree which has fast learning and retrieval ability. Considering SIFT can produce large number of features, we utilize HDR tree to implement example data training.

3.5 Procedure of Proposed Algorithm

Proposed algorithm combines the adaptive visual attention model with unit-linking PCNN to locate the ROI, which is the least square that contains vehicles. SIFT features are extracted from ROI and then are input into the trained HDR tree for vehicle detection to determine the vehicle's final location. The procedure is shown in Fig.1.

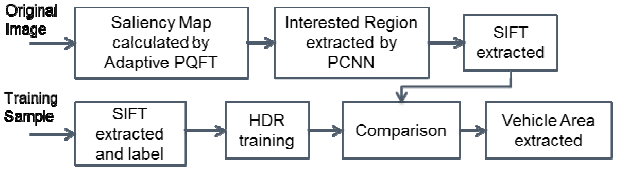


Fig. 1. Procedure of proposed algorithm

4 Experimental Results

The proposed algorithm is implemented on our constructed remote sensing image database and compared with Zheng's [1] and Thomas's algorithm [8]. We shot 112 desert images around the world of size 600*800 and resolution 0.6m*0.6m.

We calculate weight for each channel in adaptive visual attention model from training samples: $w_1 = 0.0114, w_2 = 0.0004, w_3 = 0.0147, w_4 = 0.9736$. All the programs are coded by Matlab R2010a and run in Windows 7 environment. The configuration of the operating system is Inter Core i7-2600 CPU, 3.8GHz frequency and 16GB ram.

4.1 Comparisons of Visual Attention Models

Our adaptive PQFT model is firstly compared with other visual attention models. The visual attention models compared with the proposed on include: GBVS, STB, AIM and PQFT. Fig.2 shows these experimental results.

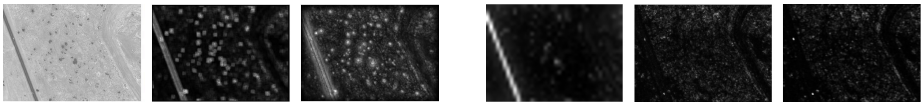


Fig. 2. Comparisons of different Visual Attention Model

Fig.2 illustrates that the saliency map produced by adaptive PQFT has distinct levels: background is dark and uniform white vehicles on the road have higher intensity value and stronger saliency.

Table 1. Comparisons of different Visual Attention Model

Model	AIM	STB	GBVS	PQFT	Adaptive PQFT
Time	22.79s	2.33s	2.41s	0.327s	0.136s

Table 1 show that Adaptive PQFT is much faster than any other visual attention models. Because this model utilizes quaternion Fourier transform that can be decomposed to traditional two-dimensional Fast Fourier Transform added to a simple linear transform, thus its calculation has high speed. Meanwhile, Adaptive PQFT utilizes bilinear interpolation to reduce the amount of data in Fourier transform.

4.2 Comparisons of Processing Effects

Our algorithm utilizes the pulse-coupled neural network to process saliency map in order to remove noise and background interference. The results are shown in Fig.3.

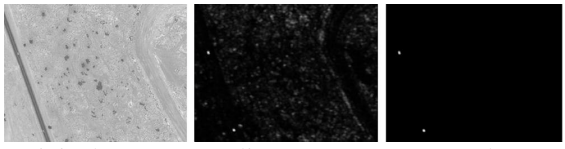


Fig. 3. Processing results with PCNN and without PCNN

After processed by pulse-couple neural network, a large number of scattered noise are removed that we get saliency map with distinct level and simple background.

4.3 Vehicle Detection Effect

As Fig.4 shows, proposed algorithm can detect all the vehicles correctly. Especially in the third subfigure, although there are three different vehicles running on two separate roads, our algorithm can detect all of them correctly. Through the test of our database, proposed algorithm detects vehicles in the desert effectively and in fast speed.

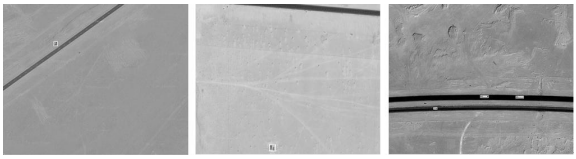


Fig. 4. Several samples' experimental results of proposed method

4.4 Comparisons with the Other Two Related Algorithms

The algorithms proposed by Zheng [1] and Thomas [8] are compared with our algorithm. In Fig.5, we show the results of these two methods using the above figures.

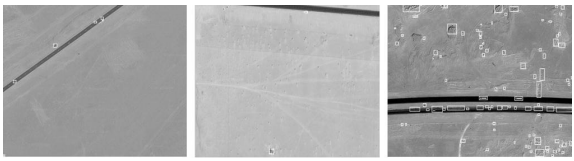


Fig. 5. Examples of Zheng's method [1] (middle one) and Thomas's method [8] (left and right ones)

Fig.5 illustrates that the false alarm problem is very serious, which many parts of the complex background are detected as vehicle regions.

Assuming m is the number of test images, n the number of images having false alarm region and y the number of images having detected all the vehicle region(allowed to have false alarm region), we can define the false alarm rate $R_{FP} = (\mathrm{n}/\mathrm{m}) \times 100\%$ and the detection rate $R_{\mathrm{RF}} = (\mathrm{y}/\mathrm{m}) \times 100\%$.

Table 2. Comparisons of different methods

Algorithm	Detection rate	False Alarm rate	Time
Zheng[1]	88.4%	67.3%	0.188s
Thomas[8]	78.8%	71.2%	6.5s
Proposed Method	96.5%	10.5%	0.371s

Table.2 illustrates that the proposed algorithm has evident advantages on detection rate and false alarm rate. On time cost, the proposed algorithm is similar to Zheng's [1] but improves a lot compared to Thomas's. Thomas's algorithm has low detection rate and also much time cost.

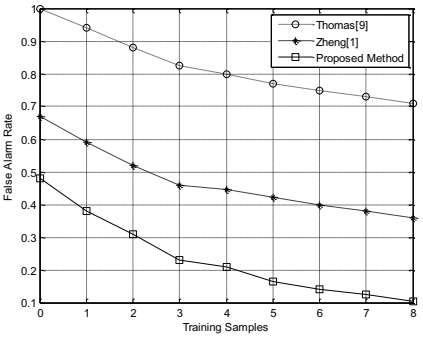


Fig. 6. Relation of Training Samples and False Alarm Rate

This paper further compares the proposed algorithm with Zheng's [1] and Thomas's [8] on the false alarm rate with different training samples. The result is shown in Fig.6. Experimental results demonstrates that the proposed algorithm has lower false alarm rate and the speed of improvement by training samples is fast which means it can achieve a low false alarm rate with only a few training samples.

5 Conclusion

In this paper, we propose a desert vehicle detection algorithm that combines the visual attention model with pulse coupled neural network. The adaptive PQFT model is combined with PCNN to locate the regions of interest which may contains vehicles. SIFT features are extracted from regions of interest and input to the trained HDR tree for classification. Experimental results show that proposed method has faster speed, higher detection rate and lower false alarm rate compared with other two works, which could meet the requirements of desert rescue.

Acknowledgments. This work was supported in part by National Natural Science Foundation of China under grant 61371148 and Shanghai National Natural Science Foundation under grant 12ZR1402500.

References

1. Zheng, Z.Z., Wang, X.T., Zhou, G.Q., et al.: Vehicle detection based on morphology from highway aerial images. In: 2012 IEEE International Conference on Geoscience and Remote Sensing Symposium (IGARSS), pp. 5997–6000 (2012)

2. Ltti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(11), 1254–1259 (1998)
3. Walther, D., Koch, C.: Modeling attention to salient proto-objects. Neural Networks 19(9), 1395–1407 (2006)
4. Guo, C.L., Ma, Q., Zhang, L.M.: Spatio-temporal saliency detection using phase spectrum of quaternion Fourier transform. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
5. Johnson, J.L., Padgett, M.L.: PCNN models and applications. IEEE Transactions on Neural Networks 10(3), 480–498 (1999)
6. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
7. Hwang, W.S., Weng, J.: Hierarchical discriminant regression. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(11), 1277–1293 (2000)
8. Moranduzzo, T., Melgani, F.: A Sift-SVM for detecting cars in UAV Images. In: 2012 IEEE International Conference on Geoscience and Remote Sensing Symposium (IGARSS), pp. 6868–6871 (2012)

An Integrated Intelligent Technique for Monthly Rainfall Spatial Interpolation in the Northeast Region of Thailand

Jesada Kajornrit[1], Kok Wai Wong[2], and Chun Che Fung[3]

School of Engineering and Information Technology, Murdoch University
South Street, Murdoch, Western Australia, 6150
j_kajornrit@hotmail.com, {k.wong,l.fung}@murdoch.edu.au

Abstract. Spatial interpolation is a method to create spatial continuous surface from observed data points. Spatial interpolation is important to water management and planning because it could provide estimation of rainfall at unobserved area. This paper proposes a methodology to analyze and establish an integrated intelligent spatial interpolation model for monthly rainfall data. The proposed methodology starts with determining the optimal number of sub-regions by means of standard deviation analysis and artificial neural networks. Once the optimal number of sub-regions is determined, a Mamdani fuzzy inference system is generated by fuzzy c-means and then optimized by genetic algorithm. Four case studies were used to evaluate the accuracy of the established models and compared with trend surface analysis and artificial neural networks. The experimental results demonstrated that the proposed methodology provided reasonable interpolation accuracy and the methodology gave human understandable fuzzy rules to human analysts.

Keywords: Monthly rainfall, Spatial interpolation, Fuzzy inference system, Standard deviation, Artificial neural network, Genetic algorithm.

1 Introduction

Spatial continuous surface of rainfall variable is important to the estimation of the amount of rainfall in an entire study area [1]. In order to acquire the spatial continuous surface of rainfall data, a number of rain gauge stations are installed throughout the study area to measure the amount of rainfall. However, the number of rain gauge stations installed could be limited due to practical reasons such as difficulty in accessing the location and cost factors. Therefore, spatial interpolation process is necessary to construct a spatial continuous surface for analysis purpose [2].

Spatial interpolation is a method used to estimate values at unsampled points by using the values from neighbouring sampled points [3]. In general, spatial interpolation process can be divided into global and local methods [4]. In the local method, spatial continuous surface is created from weighting method such as deterministic or geostatistic approaches [5]. In this method, a certain number of neighbouring sampled

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 384–391, 2013.

points are used for interpolation. In the global method, spatial continuous surface is created from an interpolation model, in which all sampled data points are used to calibrate the interpolation model such as Trend Surface Analysis (TSA) [3], [6] and Artificial Neural Network (ANN) [1], [3]. This paper focuses on the global method.

In terms of model establishment, ANN is capable to generalize the resultant model from the calibration data through nominated learning algorithms. Due to this reason, ANN is commonly recognized as an "easy-to-use" technique and it has become a popular technique in solving problem in hydrological discipline. However, the expression of ANN (or TSA) is data-oriented or being shown in the form of a set of parameters. Such characteristic can make them difficult for human analysts to understand and enhance, especially when the models are complex.

The understandability (or interpretability) of the model is another important issue in data-driven modeling because human analysts could gain the insight knowledge of the data through the model established. Further analysis can be enhanced if an explanation of the model established can be provided. This study therefore aims to propose an alternative methodology by integrating the advantages of ANN, fuzzy system and genetic algorithm for the analysis and establishment of rainfall spatial interpolation model for the northeastern Thailand region.

The rest of the paper is organized as follows. Section 2 describes the case study and datasets and Section 3 describes the proposed methodology. Section 4 illustrates the experimental results and analysis. Finally, Section 5 presents the conclusion.

2 Case Study and Datasets

The case study area is the northeast region of Thailand as shown in Fig. 1 which is located at latitude from 14.11°N to 18.45°N and longitude from 100.83°E to 105.63°E. This area covers about one third of total area of Thailand. This study selects four highest monthly rainfall data from year 1998 to 2001. The datasets comprise of information on the longitude (x), latitude (y) and amount of rainfall (z). Samples of information in the datasets are showed in Table 1. The datasets are normalized by linear transformation. In this study, approximately 30 percent of the rain gauge stations are randomly selected to validate the interpolation accuracy.

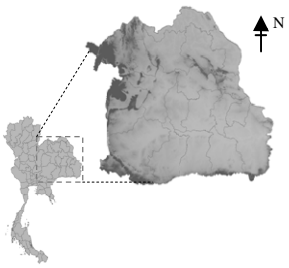


Fig. 1. The case study area locates in the northeast region of Thailand

Table 1. Information on the four datasets used in this study

Statistics	Case 1	Case 2	Case 3	Case 4
Mean	2317	2433	2756	3696
Calibration Stations	198	200	197	178
Validation Stations	80	80	80	80
Correlation	-0.047	-0.001	-0.240	-0.046

The correlation in Table 1 indicates the relationship between the amount of rainfall and the altitude of the rain gauge stations. Since the correlation values are close to zero, it could be considered that the orographic effect is not strong in the study area. So, this study does not use the altitude as one of the input to the models.

3 The Proposed Methodology

The proposed methodology consists of four steps. In the first step, the minimum number of cluster is determined. In the second step, the optimal number of cluster is selected. Once the number of cluster is selected, a Mamdani-type FIS (MFIS) is created in the third step. In the final step, the created MFIS model is optimized. Overall, step 1 and step 2 could be seen as cluster analysis, whereas step 3 and step 4 could be seen as model establishment and optimization. Fig. 2 illustrates an overview of the proposed methodology.

The first step in the proposed methodology is to determine the minimum number of clusters (or sub-regions) for the spatial data. Since the spatial data may contain uncertainty, fuzzy c-means (FCM) clustering technique is used. In the work of [7], the minimum number is determined by analyzing the standard deviation of spatial data. In this method, calibration data (x, y, z) are clustered in to n clusters where $2 \leq n \leq C_{max}$. Next, find the proportion, $P_n = E(SD_i) / n$ for each n where $1 \leq i \leq n$ and SD_i is the standard deviation of the z value in cluster i. Then, calculate the difference, $D_n = P_{n-1} - P_n$ and plot D_n against the number of cluster. The minimum number of cluster could be counted from the point when the decrease of D_n becomes stable.

The second step is to select the optimal number of cluster. Since the problem is unsupervised in which the real answer is not known. This study used two validation methods at the same time to select the optimal number of cluster. In the first method, Tutmez et al. [8] suggested that the optimal number of cluster for spatial data could be determined by

$$Minimize\ n_c \text{ under, } Std[z(x)] \approx Std[z(c)] \quad (1)$$

where n_c is the optimal number of cluster, Std is the standard deviation, $z(x)$ are the observed values of the dataset and $z(c)$ are the observed values at the cluster centers. Based on this criterion the numbers of cluster are plotted against $Std[z(c)]$. The number of clusters satisfying constrain (1) is retained as the optimal number.

For the second method, the training performance of Back-propagation Neural Network (BPNN) has been considered [7]. In this method calibration data (x, y, z) are

clustered in to n clusters where $2 \leq n \leq C_{max}$. After that, for each n, the trainingset (network's input is the calibration data and network's target is the assigned number of cluster) is used to train a group of BPNNs. The BPNNs in the group are one hidden layer BPNN with various numbers of hidden nodes. The performance, $E_n = Perf_n / n$ are calculated for each n and $Perf_n$ is the average training performance of the group of the BPNNs. The lowest E_n indicates the most appropriate number of clusters.

In the third step, once the number of cluster is determined, MFIS is generated from the calibration data by using FCM. The Membership Function (MF) used is a Gaussian function because it provides smooth surfaces and has low degree of freedom [9]. After the MFIS has been generated, it is optimized by the Genetic Algorithm (GA) technique. The chromosome of the algorithm consists of the sequence of input 1, input 2 and output respectively. In turn, the input and output are the sequence of MF which consists of two parameters (sigma and center). The fitness function is the minimize sum square error between observed value (z) and interpolated value (z') of calibration data and it is given as

$$SSE = \sum_{i=1}^{S} (z_i' - z_i)^2 \tag{2}$$

The important point of optimization is how to control the diversity of individuals [10]. In this process, the MFIS parameters are allowed to vary in certain controlled ranges based on the hypothesis that MFIS generated from cluster analysis should be near to the optimal solution. Let α and β be user-defined control parameters, the center (c) parameters are allowed to vary in the range of $[c - \alpha, c + \alpha]$ and the sigma (σ) parameters are allowed to vary within the range of $[\sigma - \beta, \sigma + \beta]$.

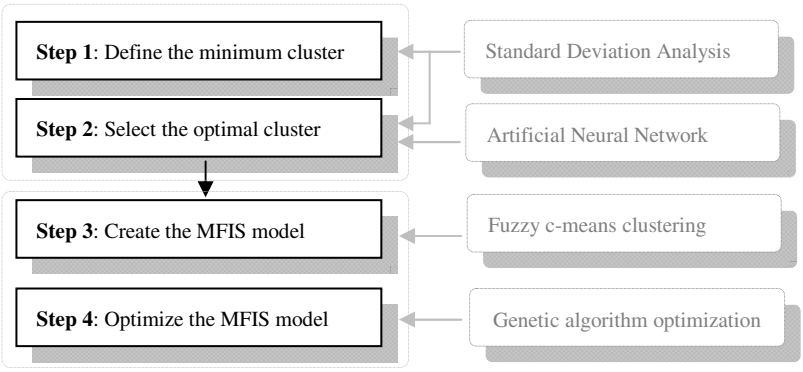


Fig. 2. An overview of the proposed integrated intelligent technique

4 Experimental Results and Analysis

In this study, interpolation accuracy of the proposed model, Integrated Intelligent Technique (IIT), was compared to TSA [3], BPNN [1], [3] and Radial Basis Function Network (RBFN) [11]. For TSA, third order polynomial equation is normally used because this order is appropriate for real-world data, which has both hill and valley

surfaces [6] and it was therefore adopted in this study. For BPNN and RBFN, this study selected the model's parameters from *K*-folds cross-validation method [13]. The calibration data were divided into five partitions (or about 20%). For each partition n, $1 \leq n \leq 5$, calibrate model with other partitions and use partition n for validation. The lowest average error from all n is used to indicate the model's parameters. For BPNN the optimal number of hidden nodes/epoch obtained from case 1 to 4 were 8/20, 7/20, 10/15 and 8 /15 respectively. For RBFN the number of hidden node were aligned with to the number of calibration data [11]. The optimal spread parameters [12] from case 1 to case 4 were 5.5, 5.0, 6.0 and 4.0.

According to Fig. 3, *in case 1*, the selected cluster was 7 because D_n became stable at $n = 7$ and the difference between *Std(z(x))* and *Std(z(c))* showed small variation after $n = 6$ (Fig. 3a). Furthermore, E_n showed the smallest value at $n = 7$ (Fig. 3b). *In case 2*, the selected cluster was 7 because D_n was stabilized after $n = 6$ and the difference of *Std(z(x))* and *Std(z(c))* showed a small variation after $n = 5$ (Fig. 3c), E_n showed relatively low value at $n = 7$ (Fig. 3d). Although E_n at $n = 7$ did not show lower value than at $n = 9$, choosing a small number of cluster could avoid making the model too complicated. Furthermore, the difference between *Std(z(x))* and *Std(z(c))* at $n = 7$ was smaller than $n = 9$. *In case 3*, D_n became stable at $n = 8$ and the difference between *Std(z(x))* and *Std(z(c))* shifted up at $n = 6$ and a small variation was observed after $n = 8$ (Fig. 3e). Therefore, $n = 8$ was selected as the cluster number. The lowest E_n took place at $n = 8$ (Fig. 3f). In *case 4,*D_n became stable at $n = 7$ and the minimum difference between *Std(z(x))* and *Std(z(c))* took place at $n = 8$ (Fig. 3g). Therefore, $n = 8$ was selected as the cluster number. At $n = 8$, it also showed the lowest value of E_n (Fig. 3h).

Once the number of cluster is determined, the MFIS is generated from the FCM technique. The number of fuzzy rules was aligned with the number of clusters. In the optimization process, the user parameter α and β were set to 0.1 and 0.05 respectively. These values were approximately 10 percent of the universe of discourse as stated before that the MFIS parameters were allowed to vary in certain controlled ranges. In this study, the number of population of GA was set to 200 (or about four times of the number of gene in the chromosome) to ensure that there are at least four individuals for each parameter. The number of generation was set to 150, where the SSE became stable and did not show any more improvement. The results are shown in Table 2.

In term of the Relative Mean Error (RME), all models showed consistent results. No suspicious of bias event was indicated. The average of the absolute RME is shown in the row Avg (i.e. average) as shown in Table 2. The models that provided better results are ranked in the following sequence: BPNN > IIT = RBFN > TSA. In terms of Relative Mean Absolute Error (RMAE) and Relative Root Mean Square Error (RRMSE), the average accuracies ranked in the order of IIT > RBFN > BPNN > TSA. However, BPNN and TSA provided almost equal interpolation accuracy. In term of Correlation Coefficient (R) the average accuracy have similar patterns as IIT > RBFN > BPNN > TSA. In the row Imp (i.e. improvement) in Table 2, the values are the percentage improvement based on TSA.

With respect to overall results, IIT can be considered as a good alternative interpolator for the Thailand northeast region because the results indicated that it provided acceptable interpolation accuracy in general. However, the objective of the proposed methodology is not only to provide an accurate interpolator for the Thailand region

but also to provide interpretable model for human analysts. As expressions in fuzzy rules are closer to human reasoning, analysts could understand how the model performed the interpolation. An example of fuzzy rules and its MFs are shown in Fig. 4. In the clustering analysis, this study made use of the standard deviation analysis to validate the number of clusters. This method is reasonable in terms of statistical point of view. Furthermore, BPNN analysis can assist the decision process with improved confidence.

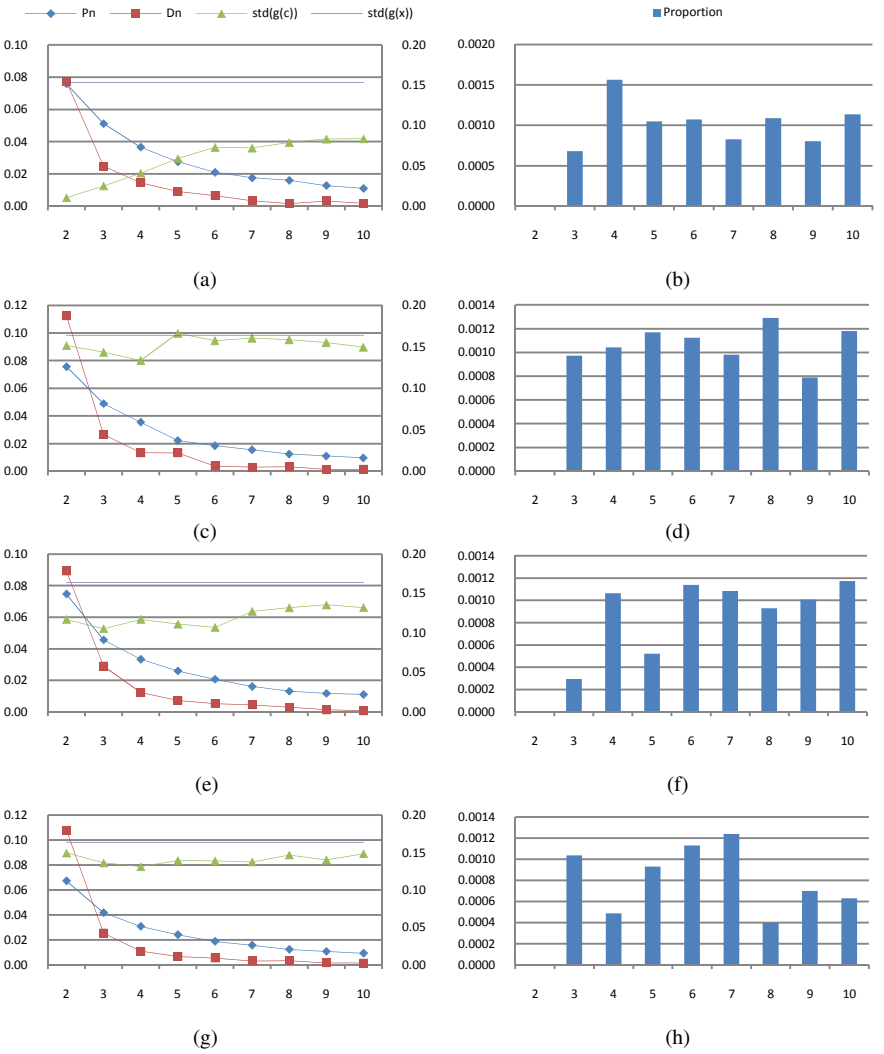


Fig. 3. The results from clustering analysis. (Note that, in Fig. a, c, e and g, the left-hand axis is for P_n and D_n values, and the right-hand axis is for *std(g(c))* and *std(g(x))* values

Table 2. RME, RMAE, RRMSE and R of validation data

	Relative Mean Error				Relative Mean Absolute Error			
Case	TSA	BPNN	RBFN	IIT	TSA	BPNN	RBFN	IIT
1	0.025	0.000	0.023	0.013	0.279	0.300	0.286	0.284
2	0.077	0.049	0.060	0.049	0.253	0.236	0.227	0.224
3	-0.008	-0.011	-0.001	-0.013	0.282	0.276	0.266	0.267
4	-0.031	-0.028	-0.019	-0.023	0.256	0.254	0.260	0.233
Avg	0.036	0.022	0.026	0.025	0.268	0.267	0.260	0.252
Imp	-	38.11	27.09	30.89	-	0.356	2.940	5.822

	Relative Root Mean Square Error				Correlation Coefficient			
Case	TSA	BPNN	RBFN	IIT	TSA	BPNN	RBFN	IIT
1	0.368	0.368	0.361	0.366	0.357	0.383	0.404	0.370
2	0.327	0.318	0.309	0.290	0.674	0.691	0.712	0.749
3	0.367	0.365	0.353	0.353	0.461	0.478	0.515	0.513
4	0.353	0.349	0.345	0.326	0.712	0.720	0.725	0.760
Avg	0.354	0.350	0.342	0.334	0.551	0.568	0.589	0.598
Imp	-	1.081	3.404	5.689	-	3.06	6.86	8.493

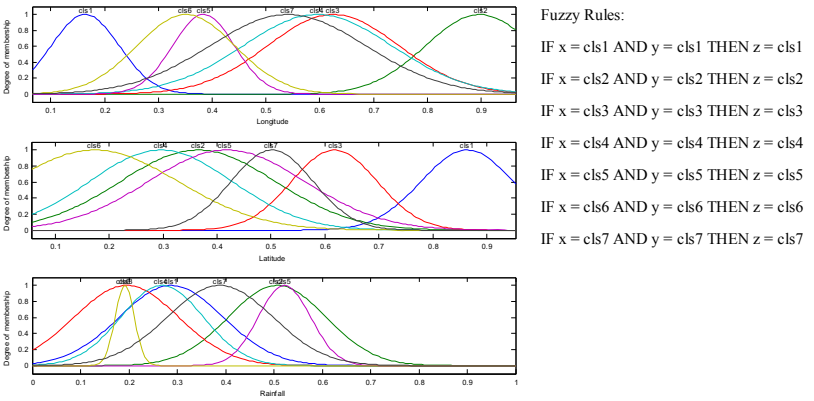


Fig. 4. An example of fuzzy rules and its membership functions (case 1)

5 Conclusions

This study proposes a methodology to analyze and establish an integrated intelligent spatial interpolation model for monthly rainfall data in the northeast Thailand. The proposed methodology starts with cluster analysis to determine the optimal number of sub-regions. Once the optimal number of sub-regions is determined, a Mamdani-type fuzzy inference system is generated. The generated fuzzy inference system is then optimized by using a genetic algorithm approach. The proposed methodology has been experimented with the monthly rainfall data in the northeast region of Thailand.

Four case studies were used to evaluate the interpolation accuracy and compared to commonly-used interpolation models. The experimental results demonstrated that the proposed model was capable to generate acceptable interpolation accuracy. Furthermore, the proposed methodology provided interpolation models that are human understandable through the use of fuzzy rule.

Reference

1. Zhang, Q., Wang, C.: Integrated application of artificial neural network and genetic algorithm to the spatial interpoation of rainfall. In: Proc. 4th International Conference of Natural Computation, pp. 516–520 (2008)
2. Li, J., Heap, A.D., Potter, A., Daniell, J.J.: Application of machine learning methods to spatial interpolation of environmental variables. Environ. Modell. Softw. 26, 1647–1659 (2011)
3. Kajornrit, J., Wong, K.W., Fung, C.C.: Estimation of missing rainfall data in northeast region of Thailand using spatial interpolation methods. AJIIPS 13(1), 21–30 (2011)
4. Burrough, P., McDonnell, R.: Principles of geographical information systems. Oxford University Press, New York (1998)
5. Li, J., Heap, A.D.: A review of spatial interpolation methods for environmental scientists, geoscience Australia, record 2008/23 (2008)
6. Chang, T.: Introduction to geographic information systems, 3rd edn. McGraw-Hill, Singapore (2006)
7. Kajornrit, J., Wong, K.W.: Cluster validation methods for localization of spatial rainfall data in the northeast region of Thailand. In: Proc. 12th International Conference on Machine Learning and Cybernetics (2013)
8. Tutmez, B., Tercan, A.E., Kaymak, U.: Fuzzy modeling for reserve estimation based on spatial variability. Mathematical Geology 39(1) (2007)
9. Hameed, I.A.: Using Gaussian membership functions for improving the reliability and robustness of students' evaluation systems. Expert Syst. Appl. 38, 7135–7142 (2011)
10. Matlab Optimization toolbox™– User's guide R2008b, Matlab
11. Lee, S., Cho, S., Wong, P.M.: Rainfall prediction using artificial neural networks. GIDA 2(2), 233–242 (1998)
12. Matlab Neural networks toolbox™ – User's guide R2008b, Matlab
13. Gilardi, N., Bengio, S.: Local machine learning models for spatial data analysis. GIDA 4(1), 11–28 (2000)

GPU Accelerated Spherical K-Means Training

Yi Xiao[1], Ruibin Feng[1], Chi-Sing Leung[1], and Pui Fai Sum[2]

[1] Dept. of Electronic Engineering, City University of Hong Kong, Hong Kong
[2] Institute of Technology Management, National Chung Hsing University
yixiao1984@gmail.com, rfeng4-c@my.cityu.edu.hk, eeleungc@cityu.edu.hk, pfsum@yahoo.com.hk

Abstract. The spherical K-means algorithm is frequently used in high-dimensional data clustering. Although there are some GPU algorithms for K-means training, their implementations suffer from a large amount of data transfer between CPU and GPU, and a large number of rendering passes. By utilizing the random write ability of vertex shaders, we can reduce the overheads mentioned above. However, this vertex shader based approach can handle low dimensional data only. This paper presents a GPU-based training implementation for spherical K-means for high dimensional data. We utilizes the feature of geometry shaders to generate new vertices to handle high-dimensional data.

Keywords: GPU, Spherica k-means, Geometry shaders.

1 Introduction

The spherical K-means algorithm, which uses the cosine dissimilarity instead of Euclidean distance, is a special case of the K-means algorithm. It is frequently used in high-dimensional data clustering applications, such as document clustering [1] and market basket analysis [2]. Due to the large data volume in these applications, the training speed of spherical K-means is an important concern.

Nowadays, graphics processing units (GPUs) become popular computing tools [3] for non-graphics applications, such as wavelet transform [4] and neural network simulation [5]. Takizawa et al. [6] proposed a CPU–GPU co-processing solution for K-means training. One of drawbacks of Takizawa's approach is that the data transfer between CPU and GPU is very large. Also the large number of rendering passes within a training iteration is very large. Xiao et al. [7] solved the above problems by investigating the random write ability of vertex shaders. In their method, each training vector is held by a vertex. Since the number of attributes in a vertex is limited and the number of writable textures in a fragment shader is limited, their implementation can handle low dimensional data only.

This paper addresses this limitation based on geometry shaders which have the ability to generate vertices. In our approach, training vectors are stored in a texture. The codebook is stored in eight textures. Each training vector is associated with a vertex (calling *training vector vertex*) but the vertex stores

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 392–399, 2013.

the training vector index only. A vertex shader finds out the winner for each training vector, and then stores the winner index and the training vector index in a processed vertex (calling *winner vertex*). All the winner vertices are then passed to a geometry shader. For each winner vertex, the geometry shader creates a number of *sub–winner vertices*. Each sub–winner corresponds to updating 32 elements in a codevector. In general, a vertex can generate 1024 vertices in a geometry shader. Therefore, our algorithm can handle data vectors with very large dimension.

2 Background

Spherical K-means: The spherical K-means partitions a set of training vectors $\mathbb{D} = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{N-1}\}$ in $\Re^k$ into M clusters $\Omega_i, i = 0, \cdots, M-1$, whose codevectors (centers of clusters) are given by $\mathbb{Y} = \{\boldsymbol{c}_0, \cdots, \boldsymbol{c}_{M-1}\}$. In training, the objective is to minimize the objective function: $E = \frac{1}{N}\sum_i \sum_{\boldsymbol{x}_j \in \Omega_i}(1 - \cos(\boldsymbol{x}_j, \boldsymbol{c}_i))$, where $\cos(\boldsymbol{x}_j, \boldsymbol{c}_i) = \frac{<\boldsymbol{x}_j, \boldsymbol{c}_i>}{\|\boldsymbol{x}_j\|\|\boldsymbol{c}_i\|}$. The collection of codevectors $\mathbb{Y}$ is called a codebook. Given the data set $\mathbb{D}$ and the initial codebook $\mathbb{Y}(0)$, the training process can be summarized as follows.

1. $t = 0$.
2. Set $\Omega_i = \varnothing$, for all $i = 0, \cdots, M-1$.
3. For each training vector $\boldsymbol{x}_j$, find out the winner codevector $\boldsymbol{c}_{i^*}$, where $i^* = \arg\min_{i \in \{0,\cdots,M-1\}}(1 - \frac{<\boldsymbol{x}_j, \boldsymbol{c}_i>}{\|\boldsymbol{x}_j\|\|\boldsymbol{c}_i\|})$. Put $\boldsymbol{x}_j$ into the subset Ω_{i^*}.
4. Update the codevectors: $\boldsymbol{c}_i(t+1) = \frac{\sum_{\boldsymbol{x}_j \in \Omega_i} \boldsymbol{x}_j}{\|\sum_{\boldsymbol{x}_j \in \Omega_i} \boldsymbol{x}_j\|}, \forall i$.
5. Set t=t+1 and go to Step 2.

The above iterative procedure repeats until the objective value E is less than a threshold, or the number of iterations has reached a pre-defined number.

GPU : In the GPU pipeline, there are three kinds of programs (shaders): vertex shader, geometry shader and fragment shader. A vertex shader is response for transforming properties of the input vertices. The outputs of the vertex shader are then assembled to geometries, which can be vertices, lines or triangles. A geometry shader performs the operations on these geometries. It can generate new graphics primitives, such as points, lines, and vertices. The rasterization then outputs fragments (pixels) based on the received geometries. A fragment shader deals with the rasterized fragments. It computes color and other attributes of each fragment.

3 Our Method

Our implementation can be considered as a multi-client-multi-server model. Each training vector is a client and each codevector is a server. Each client finds its

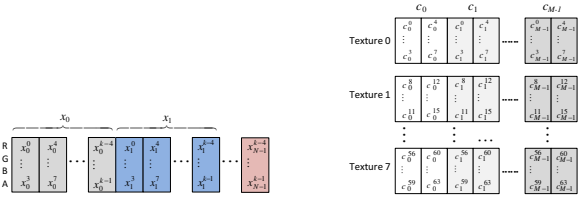
winner codevector and sends itself to the server. The server accumulates the training vectors arrived. After all clients are served, each server normalizes its accumulated vector. Let N be the number of training vectors, let M be the number of clusters, and let k be the dimension of the vectors. Without loss of generality, we assume that k can be divided by 4.

3.1 Initialization

The training vectors are linearly stored in the texture memory, as shown in Fig. 1(a). This texture is called *training data texture*. The resolution of the texture is $1 \times \frac{k}{4}N$. Each texel stores 4 elements of a training vector. To access the training vectors on the GPU, we prepare a list of N vertices, each of which stores the index of a training vector(from 0 to $N-1$). The list is stored on the GPU as a *display list*. These vertices are called *training vector vertices*.

A codebook is organized as a collection of T textures. Nowadays, the maximum number of textures that can be written simultaneously is 8. So, we use $T = 8$ textures. The logical structure of a codebook on the GPU for $k = 64$ is shown in Fig. 1(b). For large M and k, 2D textures are used. The resolution of each texture is equal to $1 \times \frac{k}{4T}M = 1 \times \frac{k}{32}M$. Each texel corresponds to 4 elements of a codevector, and then each texel position corresponds to 32 elements of a codevector. In the implementation, there are two codebooks. One stores the current codebook. The initial current codebook is transferred to the texture memory. The other one is the temporary codebook which contains the summation of training vectors arrived in clusters. That means, there are 16 textures for codebooks.

To facilitate updating the codebook, we prepare another list of M vertices, each of which stores the indices of codevectors (from 0 to $M-1$). The list is stored as a display list. After the textures and display lists are initialized, we then use 6 shaders to implement the spherical k-means training. The flow diagram of one training iteration is shown in Figure 2. The whole implementation consists of two stages. In the first stage, we perform two tasks. We search the winner codevector and sum the training vectors in clusters. In the second stage, we normalize the codevectors.



(a) Texture for training vectors (b) Textures for codevectors

Fig. 1. (a) The logical structure of the training vectors stored in GPU. Each grid denotes a 4 channel texel. (b) The logical structure of codevectors stored in GPU for dimension $k = 64$.

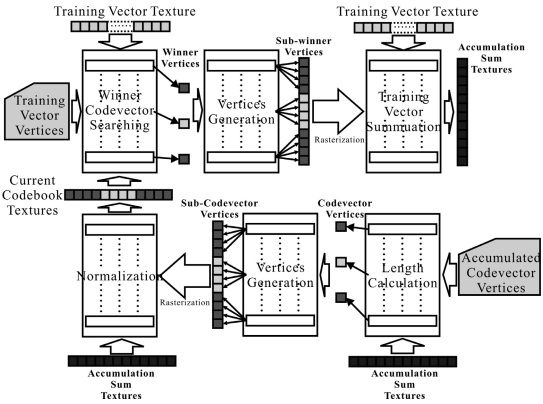


Fig. 2. The overview of our algorithm

3.2 Winner Codevector Searching and Training Vector Summation

In the first part, the GPU searches the winner codevector for each training vector. It then puts the training vector into the corresponding cluster, and calculates the summation of all the training vectors received in each cluster. In the implementation, we use a vertex shader, a geometry shader and a fragment shader to perform the above tasks. As mentioned in the above, there are 16 textures for codebooks. One set of 8 textures (for read) holds the current codebook. The other set of 8 textures (for write) holds the accumulation sum of training vectors in clusters, which are initially set to zero. The write ability of textures is realized by using frame buffer objects (FBOs).

Winner Codevector Searching (vertex shader): A vertex shader is used to search the winner codevector for each training vector. It then passes the information of the training vector and the winner codevector to the geometry shader. A display list contains a number of vertices. Each of them stores a training vector index as the position information of the vertex. For ease of read, we call these vertices as *training vector vertices*. For each index (*training vector vertex*), the vertex shader finds its winner codevector by reading its corresponding training vector from the training vector texture and the current codebook textures. The outputs of this shader are vertices.

For each training vector vertex, the shader generates an output vertex that contains the winner codevector index and the training vector index. The winner index is transformed into the texture coordinate of the winner codevector. The coordinate is then set as the position information of the output vertex. Meanwhile, the training vector index and the winner index are set as attributes of the vertex. For ease of read, we call those output vertices as *winner vertices*. For instance, for $k = 64$ and $M = 256$, if the training vector index is j and the winner index is i^*, then the corresponding winner vertex contains the following messages: $\{pos = \frac{\frac{k}{32} \times i^* + 0.5}{\frac{k}{32} \times M}, j, i^*\} = \{pos = \frac{2 \times i^* + 0.5}{512}, j, i^*\}$. Note that a fragment shader in GPU can only access 8 "writing" textures. Hence the vector

dimension in the training is limited to 32 only, if we directly use winner vertices to generate fragments.

Vertices Generation (geometry shader): The geometry shader receives the winner vertices. For each winner vertex, the geometry shader generates $\frac{k}{32}$ output vertices based on the received messages. Each of the output vertices corresponds to 32 elements in the winner codevector. For ease of read, we call those output vertices as *sub-winner vertices*. The $\frac{k}{32}$ sub-winner vertices of the winner codevector is used to accumulate a training vector to a winner cluster. Each of them handles 32 elements.

A sub-winner vertex contains three messages: the texture coordinate of the 32 elements concerned, the training vector index j, and an element index which indicates which set of 32 elements is used. For $k = 64$ and $M = 256$, if an incoming winner vertex contains the following messages: $\{pos = \frac{2\times i^* + 0.5}{512}, j, i^*\}$, the geometry shader generates two sub-winner vertices:SWV_0 and SWV_1. Their messages are $SWV_0 = \{pos = \frac{2\times i^* + 0 + 0.5}{512}, j, 0\}$ and $SWV_1 = \{pos = \frac{2\times i^* + 1 + 0.5}{512}, j, 1\}$. Note that $\{0, 1\}$ here are the element indices which the vertex handles.

Training Vectors Summation (fragment shader): In our setting, each sub-winner vertex generates a fragment after rasterization. The texture coordinate of the fragment is given by the sub-winner vertex. Since each fragment contains its training vector index and an element index, for each fragment the shader can directly read the corresponding 32 elements from the training vector texture. These elements are set as the colors of the fragment, and output to the 8 "writing" accumulation sum textures. This is implemented by the multi-rendering targets functionality of GPU. With the blending functionality enabled, the training vectors which belong to each cluster are then accumulated in the 8 "writing" accumulation sum textures. After all the fragment are processed, the writing textures hold the summation of all the training vectors received in each cluster.

3.3 Normalization Stage

In this stage, the GPU calculates the length of each accumulated codevector. By the normalizing the accumulated codevectors with the lengths, the GPU obtains the updated codebook. In the implementation, we also use a vertex shader, a geometry shader and a fragment shader to perform the above tasks. In this stage, the 8 current codebook textures become "writing"set, and the 8 accumulation sum textures becomes "reading" set. In this stage, we also have a display list containing the indices of accumulated codevectors. Each input vertex in the list stores the index of a accumulated codevector as the position information of the vertex. For ease of read, we call these vertices as *accumulated codevector vertices*.

Length Calculation (vertex shader): This vertex shader is response for calculating the lengths of the accumulated codevectors, and passing the lengths to the geometry shader. By drawing the display list containing the indices of codevectors, the indices (as the position information of the vertices) are passed to the vertex shader. For each index, the vertex shader reads its accumulated codevector from the "reading" textures and calculates its length. For each

index (accumulated codevector vertices), the shader outputs a vertex. For ease of read, we call the output vertex as *codevector vertex*. The index of the codevector is set as the position information of the output vertex. Meanwhile, the calculated length is set as an attribute of the vertex. If the codevector index is i and the calculated length is 4567, then the codevector vertex contains the following message: $\{pos = \frac{\frac{k}{32} \times i + 0.5}{\frac{k}{32} \times M}, i, 4567\} = \{pos = \frac{2 \times i + 0.5}{512}, i, 4567\}$.

Table 1. Speeds of different approaches. The speed is measured in number of iterations per second. The vector dimension is 512. N: the number of training vectors, and M: the number of codevectors.

N		M=32	M=64	M=128	M=256	M=512
4096	CPU Method	7.204	3.617	0.594	0.222	0.046
	Takizawa's Method	28.32	14.70	7.168	3.608	1.810
	Our Method	**149.25**	**70.42**	**41.84**	**15.33**	**8.733**
16384	CPU Method	1.771	0.897	0.224	0.074	0.034
	Takizawa's Method	10.07	5.235	2.628	1.328	0.667
	Our Method	**32.05**	**21.50**	**10.23**	**5.20**	**2.880**
65536	CPU Method	0.449	0.226	0.113	0.056	0.016
	Takizawa's Method	2.635	1.367	0.695	0.351	0.175
	Our Method	**9.025**	**5.120**	**2.840**	**1.489**	**0.736**

Vertices Generation (geometry shader): Thereafter, the geometry shader receives the information of vertices. Based on each input vertex, the shader emits $\frac{k}{32}$ vertices, each of which corresponds to 32 elements in the codevector. For ease of read, we call those output vertices as *sub-codevector vertices*. A sub-winner vertex contains two messages: the texture coordinate of the 32 elements concerned and the length of the codevector. For $k = 64$, $N = 65536$, and $M = 256$, if an incoming codevector vertex contains the following messages: $\{coord = \frac{2 \times i + 0.5}{512}, i, 4567\}$, the geometry shader generates two sub-codevector vertices:SCV_0 and SCV_1.

Their messages are $SCV_0 = \{pos = \frac{2 \times i + 0 + 0.5}{512}, \frac{2 \times i + 0 + 0.5}{512}, 4567\}$ and $SCV_1 = \{pos = \frac{2 \times i + 1 + 0.5}{512}, \frac{2 \times i + 1 + 0.5}{512}, 4567\}$. Note that the texture coordinate appears twice. One is set as the position information of the sub-codevector vertex. The position information is for GPU to determine the texture coordinate of the "writing" textures in the later stage. The other one is set as the attribute of the sub-codevector vertex, and it is for GPU to determine the texture coordinate of the "reading" textures in the later stage.

Normalization (fragment shader): In our setting, each sub-codevector vertex generates a fragment after rasterization. The texture coordinate of the fragment is given by the position information from the sub-codevector vertex. The length of the codevector is also obtained. The fragment shader can directly read the corresponding 32 elements of the accumulation sum from the "reading" textures. These elements are then divided by the length. The normalized vales are output

to the "writing" textures. Thereafter, the "writing" textures will store the new codebook, which will be used as "reading" textures in the next iteration.

4 Experiments and Discussion

Three approaches are considered. They are our approach, the Takizawa's approach, and the CPU approach. All results are obtained using a PC with its configuration: Intel i7-3770 CPU, 16GB Memory, GeForce GTX 690 Display Card, 2GB Video Memory, PCI-Express 16 X Interface, Windows 7 (64 bit), and GLSL Shader Language.

The dimension k we test are 512 and 2048. All the training vectors and initial codevectors are randomly generated unit vectors. For each dimension, we test different number of training vectors N and number of codevectors M. The training speeds for different settings, measured with iterations per seconds, are given in Tables 1-2. As we can see from the tables, the CPU method is quite slow. Our GPU method can run more than 10-50 times faster than the CPU method. For example, when $k = 512, M = 65532$, $N = 512$, our algorithm is about 46 times faster than the CPU method. Meanwhile, our GPU method is also 2 to 8 times faster than Takizawa's. The improvement is due to the reduction of the number of rendering passes, and the reduction of the amount of data transfer between CPU and GPU memories.

Table 2. Speeds of different approaches. The speed is measured in number of iterations per second. The vector dimension is 2048. N: the number of training vectors, and M: the number of codevectors

N		M=32	M=64	M=128	M=256	M=512
4096	CPU Method	1.800	0.897	0.148	0.055	0.011
	Takizawa's Method	2.396	1.209	0.584	0.292	0.145
	Our Method	**11.94**	**6.540**	**3.103**	**1.633**	**0.836**
16384	CPU Method	0.448	0.226	0.055	0.018	0.008
	Takizawa's Method	0.966	0.489	0.237	0.118	0.059
	Our Method	**2.927**	**1.269**	**0.638**	**0.334**	**0.164**
65536	CPU Method	0.112	0.056	0.028	0.014	0.004
	Takizawa's Method	0.250	0.127	0.062	0.031	0.015
	Our Method	**0.631**	**0.289**	**0.144**	**0.072**	**0.037**

5 Conclusion

In this paper, we present a novel GPU accelerated spherical K-means training for high dimensional data. Using the scattering ability of the vertex shader, we accumulate the training vector to the position of nearest codevector. We then use the geometry shader to generate extra vertices for handling the high dimensional data. With these two operations, we then accumulate the codevectors to the frame buffer with the fragment shader. Similar tricks are used in updating the codevectors. Therefore, all the operations are performed on GPU.

Acknowledgement. The work was supported by RGC Competitive Earmarked Research Grant from Hong Kong (Project No.: CityU 115612).

References

1. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Mach. Learn. 42(1-2), 143–175 (2001)
2. Tan, P.N., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. Inf. Syst. 29(4), 293–313 (2004)
3. Mark, W.R., Glanville, R.S., Akeley, K., Kilgard, M.J.: Cg: A system for programing graphics hardware in a C-like language. ACM Transactions on Graphics 22(3), 896–907 (2003)
4. Wong, T.T., Leung, C.S., Heng, P.A., Wang, J.: Discrete wavelet transform on consumer-level graphics hardware. IEEE Trans. on Multimedia 9(3), 668–673 (2007)
5. Ho, T.Y., Lam, P.M., Leung, C.S.: Parallelization of cellular neural networks on GPU. Pattern Recogn. 41(8), 2684–2692 (2008)
6. Takizawa, H., Kobayashi, K.: Hierarchical parallel processing of large scale data clustering on a PC cluster with GPU co-processing. J. Supercomput. 36, 219–234 (2006)
7. Xiao, Y., Leung, C.S., Ho, T.Y., Lam, P.M.: A GPU implementation for LBG and SOM training. Neural Computing and Applications 20(7), 1035–1042 (2011)

Online Learning: Searching for the Best Forgetting Strategy under Concept Drift

Ghazal Jaber[1,2,3], Antoine Cornuéjols[1,2], and Philippe Tarroux[3]

[1] AgroParisTech, UMR 518 MIA, F-75005 Paris, France
[2] INRA, UMR 518 MIA, F-75005 Paris, France
[3] Université de Paris-Sud, LIMSI, Bâtiment 508, F-91405 Orsay Cedex, France
{ghazal.jaber,antoine.cornuejols}@agroparistech.fr,
philippe.tarroux@limsi.fr

Abstract. Learning from data streams in the presence of concept drifts has become an important application area. When the environment changes, it is necessary to rely on on-line learning with the capability to forget outdated information. Ensemble methods have been among the most successful approaches because they do not need hard-coded and difficult to obtain prior knowledge about the changes in the environment. However, the management of the committee of experts which ultimately controls how past data is forgotten has not been thoroughly investigated so far. This paper shows the importance of the forgetting strategy by comparing several approaches. The results lead us to propose a new ensemble method which compares favorably with the well-known CDC system based on the classical "replace the worst experts" forgetting strategy.

Keywords: Online learning, ensemble methods, concept drift.

1 Introduction

Recent years have witnessed the emergence of a new research area which focuses on learning from data streams in the presence of *evolving concepts*. For instance, spam filtering systems are continuously classifying incoming emails (observation $\mathbf{x}$) into spam or non-spam (label y) depending on their content. Because of changes in the spammers' strategies, corresponding to a change of the conditional distribution function $p(y|\mathbf{x})$, the filtering systems must adapt their decision rule lest they rapidly become useless.

When learning under concept drift, one central concern is to optimize a trade-off between learning from as much data as possible, in order to get the most precise classification model, while at the same time recognizing when data points become obsolete and potentially misleading, impeding the adaptation to new trends. This is known as the *stability-plasticity dilemma*. While stability entails accumulating knowledge regarding the supposedly stationary underlying concept, plasticity, however, requires *forgetting* some or all of the old acquired knowledge in order to learn the new upcoming concept.

On-line ensemble methods have raised much interest in recent years ([1–6]). For a large part, this is due to the fact that they seem to adapt more naturally

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 400–408, 2013.

to changes in the environment than other approaches based on explicit strategies for controlling the stability-plasticity dilemma ([7]). Because of the assumed diversity of the base learners[1] in the committee, it is indeed expected that at any time some of them are ready to take over and adapt to the novelties of the environment ([8]). This diversity, however, ultimately depends on the information kept by each base learner and on the control of which learner is authorized to be part of the committee. In addition, the way the votes of the base learners are combined participates also in the overall solution to the stability-plasticity dilemma. Each of these three factors: the memory of each expert, the control of the population of experts in the committee and the weight attached to each expert in the final decision, plays a role in the way past data is taken into account by the system, what can be called the forgetting strategy.

In most ensemble methods, the first factor is implicitly governed by the second one. Each expert learns using an ever growing memory of the past data until the controller of the pool of experts decides to expel it from the committee.

In compliance with the demands of the stability-plasticity dilemma, the control strategy must be ready to introduce in the committee new base learners that will try to catch up with potential novel regularities in the environment. At the same time, it must *weaken the effect of past data that no longer represent relevant information.* There exist *two main approaches* to this problem. One is to set a threshold on the performance of the expert and to remove from the committee all experts of which the prediction performance falls below this threshold. The idea is to remove all experts that are overly biased toward obsolete regularities of the environment. This approach raises two issues. First, how to measure the prediction performance of each base learner? Second, how to set the threshold? The second family of methods does not depend on a threshold but relies instead on a perpetual renewal of the population of the committee which tends to favor a higher level of diversity. The concern here is to remove the experts that are less relevant to the current environment. Again, the question arises about the appropriate measure of performance. In addition, one must choose an insertion strategy in order to allow for the introduction of new base learners in the pool.

This paper focuses on the possible control strategies and on their impact on the performance of the system depending on the characteristics of the changing environment. We compare the two families of approaches with a special attention to the study of the deletion strategy. We do not consider the voting strategy here and keep it constant for all systems that we compare.

In the following, Section 2 describes the framework of the ensemble methods used to adapt to concept drifts while Section 3 addresses the analysis of the strategies presented above. Section 4 presents a new ensemble method using an enhanced forgetting strategy. Section 5 then reports an extensive comparison of our method with CDC ([2]) a well-known and representative ensemble method. Finally, Section 6 concludes and suggests future research directions.

[1] We use interchangeably the terms “base learner” and “expert” in this paper.

2 Ensemble Methods for On-Line Learning

Ensemble methods for on-line learning maintain a pool of base learners $\{h_t^i\}_{1 \leq i \leq N}$, each of them adapting to the new input data, and administer this pool or ensemble thanks to a deleting strategy and an insertion one. The main components of these ensemble methods are the following:

- *Learning*: each base learner in the pool continuously adapts with new incoming data until it is removed from the pool.
- *Deletion strategy*: every τ time steps, the base learners are *evaluated* on a window of size τ_{eval}. Based on the results of the evaluation, base learners might be eliminated from the committee.
- *Insertion strategy*: every τ time steps, new base learners can be created and inserted in the pool. Each new learner starts from scratch with no knowledge of the past. It is protected from possible deletion for a duration τ_{mat}.
- *Prediction*: For each new incoming instance $\mathbf{x}_t$, the prediction $\tilde{y}_t = H(\mathbf{x}_t)$ of the committee results from a *combination* of the predictions of the individual base learners $h_t(\mathbf{x}_t)$.

Variations around this general framework lead to specific algorithms ([2–6]). The remainder of this paper will be concerned with the *deletion strategy* i.e. with the base learners selected for deletion. The insertion strategy will simply replace deleted base learners by new ones in order to keep the committee size fixed. This approach is used in most current ensemble methods ([2, 3, 5, 6]).

3 Analysis of the Deletion Strategies

The deletion strategy plays a key role in the adaptation process since it allows the system to forget the memory of outdated training data. In this section, we explicitly study deletion strategies based on a threshold and deletion strategies that remove the worst base learners in the committee. For the latter approach, we compare systems based on the removal of the worst expert with systems that remove experts based on other strategies.

3.1 Deletion Strategies Using a Threshold Value

A deletion strategy based on a threshold replaces base learners in the ensemble when their *prediction record*, evaluated on the window size τ_{eval}, is below a predefined threshold θ_d. When the performance is computed as the percentage of correctly classified instances on the evaluation window, only the base learners with a classification accuracy of at least $\theta_d\%$ thus remain in the committee.

This strategy leads to different behaviors depending on the characteristics of the environment. For the sake of the analysis, let us suppose that a concept drift

occurs corresponding to a change in the label of $sev\%$ of the input space. This is called the *severity* of the concept drift ([8]). Let's suppose further that all learners in the committee have a perfect classification accuracy of 100% before the drift. We are then faced with a difficult conundrum.

If $sev << 1 - \theta_d$, the severity of the concept change is below the detection capability, and we may end up with an unchanged committee of base learners.

However, the choice of a higher threshold is loaded with two potential pitfalls. First, in case of a noisy environment, the classification accuracy of many learners may drop under the θ_d value resulting in a severely impoverished committee even though no real concept drift did happen. Second, new base learners may not be able to reach the exacting threshold before they reach the maturity age (τ_{mat}) and are therefore no longer protected from deletion.

Overall, it is difficult to set a value for a threshold without well-informed prior knowledge on the dynamics of the environment. Too low a threshold threatens the plasticity of the system, while a high one may cause havoc in the committee and prevent stability and good prediction performance. For these reasons, ensemble methods that do not rely on explicit threshold have been promoted.

3.2 Strategies That Delete the Worst Base Learner

Rather than setting a threshold for deciding which base learners to eliminate, one can encourage the diversity in the committee while preserving the best current base learners by removing the worst one every τ time steps. This should discard base learners that no longer correspond to the current state of the environment and introduce at the same time new base learners. However, a potentially vicious interaction involving the parameters τ and τ_{mat} may ruin this hope.

Let us first suppose that the period of time during which a new base learner is protected from deletion: τ_{mat} is less than τ. At each new deletion time, the newest base learner is prone to be deleted and will be if it did not have time to learn enough of the regularities in the environment. But τ cannot be too large lest the system looses any plasticity.

Suppose then that $\tau \leq \tau_{mat}$. Again the risk exists that deletion will affect only the newest learners in the committee effectively dividing the committee into a protected subset of the best and oldest base learners and a subset of the newest ones that are never able to catch up with the other ones except when the overall performance of the system has so declined that even a low prediction performance may allow a base learner to avoid elimination. And decreasing the value of τ cannot solve this problem either because if then the newly introduced base learners can survive more deletion cycles, they will still not be able to reach the performance of the established experts. Moreover, this will then tend to introduce new learners at a too high rate, threatening now to disrupt the stability and therefore the committee's performance. A question is then whether it is possible to break this poor behavior which impedes plasticity by never allowing new learners to enter the top subset.

3.3 A New Deletion Strategy Based on a Stochastic Mechanism

One can soften the "eliminate the worst learner" strategy's drawbacks by picking randomly a learner from the subset of the k ($k < N$) worst base learners. In this way, the newest base learners have a chance to learn enough of the regularities of the current environment to enter the pool of the top experts, and, at the same time, preserving the best performers. This promotes the plasticity of the system while not deteriorating its stability. The size k of the subset where base learners can be picked up to be eliminated controls the plasticity-stability trade-off.

We studied the effect of five deletion sizes (by setting the value of k) on the forgetting strategy: N, $0.75 * N$, $0.5 * N$, $0.25 * N$ and 1 (corresponding to the "always eliminate the worst base learner)". Figure 1 shows the mean classification error depending on the different deletion sizes on the datasets of the *Line*, *SineH* and *Circle* artificial problems suggested by Minku et al. [8] to evaluate drift handling methods. The parameters were set to the values: $\tau = \tau_{mat} = \tau_{eval} = 20$, and N is 10, 20 or 30. The global prediction merely uses the prediction from the current best base learner. The base learners are decision trees (as implemented in Matlab) and all the experiments start with the same random seed so that we have the same learners at the beginning of the experiments.

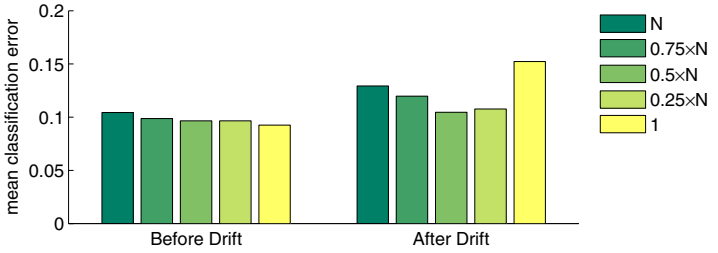


Fig. 1. The mean classification error using different deletion sizes

Case of a Stationary Environment. The deletion size $k = 1$ gives the best classification accuracy when learning in a stationary environment. The learners trained on the smallest windows are generally the ones that tend to be removed from the ensemble since they perform poorly compared to learners that have benefited from a large training set. Meanwhile, the remaining learners tune up their knowledge of the current concept, improving their classification record. By increasing k, the probability of removing a relatively good learner is also increased which hurts (to some extent) the classification accuracy of the ensemble.

Case of a Concept Drift. Increasing k increases the probability of a newly added learner to survive a deletion. A large deletion size removes most of the learners from the ensemble after a concept drift. Thus, for maximum plasticity, the best deletion size is $k = N$.

The experiments suggest that k should be small enough for stability and large enough for plasticity. With a minimum deletion size ($k = 1$), the ensemble has the lowest classification error before the drift because *stability is favored over plasticity*. With a maximum deletion size however ($k = N$) the ensemble *favors plasticity over stability* which hurts the classification performance when learning stationary concepts.

A deletion size that is half the size of the ensemble ($k = 0.5 * N$) seems to correspond to a satisfactory trade-off between plasticity and stability. It yields the lowest classification error in average, before and after the concept drift.

4 DACC

We devised DACC (*dynamic adaptation to concept changes*), an online ensemble method with adaptation to possible concept drifts.

Instead of removing the worst learner of the pool, DACC selects randomly a member from the worst half of the pool and forces it to retire. In order to control the rate of deletion, we impose all the learners to be mature before a deletion operation. Therefore, $\tau = \tau_{mat}$ time steps separates two consecutive deletions.

A learner h_{bad} belonging to the worst half of the pool survives a deletion operation with a probability

$$p = \frac{N/2 - 1}{N/2} \tag{1}$$

Each time h_{bad} escapes deletion, it is given another τ_{mat} time steps of training data before the next deletion operation. The expectation of s, the number of times h_{bad} survives a deletion operation, is:

$$\begin{aligned} E[s] &= \sum_{m=1}^{\infty} mp^m(1-p) = p(1-p)\sum_{m=1}^{\infty} mp^{m-1} \\ &= p(1-p)\frac{\mathrm{d}}{\mathrm{dp}}\left(\sum_{m=1}^{\infty} p^m\right) = p(1-p)\frac{\mathrm{d}}{\mathrm{dp}}\left(\frac{p}{1-p}\right) \\ E[s] &= \frac{p}{(1-p)} \end{aligned}$$

By replacing p with its value from equation 1, we get: $E[s] = \frac{N}{2} - 1$

Increasing the life expectancy of the relatively bad learners in the pool makes DACC less exposed to cases where new learners are expelled from pool because they didn't get enough time to improve their predictive performance. In other words, the suggested forgetting strategy is less sensitive to a high deletion rate than the *replace the worst learner* strategy. A higher deletion rate entails a faster reactivity to a potential concept drift.

The new deletion strategy creates the following behavior. In periods of stability, the top base learners, being protected from deletion, will have their prediction performance improved with time as new training data are received and learnt. Their improved performance will further keep them in the top subset, allowing

them to accumulate knowledge about the underlying stable target concept. The worst half of the pool will undergo periodic deletion operations. By constantly adding new learners, DACC is ready to any upcoming change, and this, without explicitly identifying a concept drift. Hence, young learners tend to be valuable during changing times while oldest learners are reliable in stable environments.

DACC follows the framework described in Section 2. The *evaluation procedure* simply counts the number of erroneous predictions on the last τ_{eval} time steps. The *deletion strategy* randomly selects one base learner from the worst half of the pool evaluated as above every $\tau = \tau_{mat}$ time steps. The *global prediction* merely uses the prediction from the current best base learner (a vote is applied in case of ties). An unmature learner does not contribute to the global prediction.

5 Experiments

We evaluated the mean classification error of DACC, CDC [2] and a learning system that does not handle drifting concepts. CDC differs from DACC in two major points. First, its *deletion strategy* evaluates the learners each time step (i.e. $\tau = 1$). A learner is removed if it is mature, if its evaluation record is below a threshold value, and if it is the worst learner in the ensemble. Secondly, the *global prediction* is the result of a weighted vote, where the weight of a learner reflects its evaluation record.

The experiments used artificial, semi-artificial and real datasets. The base learners were decision trees. The system that does not handle drifting concepts was a single decision tree trained on every training example received. CDC was evaluated with three thresholds: 0.6, 0.7 and 0.8.

The datasets used in the experiments are described in Table 1. The *artificial* datasets included Minku's et al. artificial problems [8]: *Circle*, *Line*, *SineH*, *SineV*, *Boolean* and *Plane*. Each problem consists of 9 datasets with different drift severity and speed levels (3 severities×3 speeds). The STAGGER [9] and FLORA [10] problems are among the pioneer artificial problems simulating drifting scenarios. FLORA consists of two datasets, with moderate and slow speeds of concept drifts, respectively. The *semi-artificial* datasets included IRIS and CAR [8], which are modified versions of the IRIS and CAR real datasets available in the UCI Machine Learning Repository [11]. The original real datasets were replicated several times and class labels were modified in order to simulate datastreams with multiple concept drifts. Finally, the *real dataset* was issued from the COLD database of the Saarbrücken laboratory [12], a benchmark for vision-based localization systems. It contains sequences of images recorded by a mobile robot under different variations of illumination and weather: sunny, cloudy and night. We worked on the dataset captured in sunny conditions. Images were first pre-processed into a 128-dimensional space using the Self-Organizing Map described in [13].

Table 1 reports the mean classification error of the different approaches along with the preset parameter values. For Minku's artificial problems, the error was averaged over the 9 different datasets of the corresponding problem. For STAGGER and FLORA problems, the error was averaged over 10 instantiations of the

Table 1. The mean classification error of the different approaches along with the predefined parameter values, with $\tau_{mat} = \tau_{eval}$. We detail the size of each dataset, the size of its feature space, the number of classes and the dataset type: A, S, R, for artificial, semi-artificial and real, respectively.

Datasets					**mean classification error**					**Settings**
name	size	#feat.	#class.	type	DACC	CDC 0.8	CDC 0.7	CDC 0.6	NoDriftH.	τ_{eval}, N
CAR	1296	6	2	S	14.66	23.47	15.14	**13.98**	40.77	20,20
IRIS	338	4	4	S	**15.33**	15.38	18.34	20.71	34.62	20,20
Circle	2000	2	2	A	**6.92**	35.66	8.09	8.73	12.56	20,20
Line	2000	2	2	A	**3.72**	8.12	4.64	7.01	10.52	20,20
Boolean	1000	3	2	A	**2.75**	3.85	4.17	4.47	17.63	20,20
SineH	2000	2	2	A	**13.0**	47.81	20.66	13.26	21.25	20,20
SineV	2000	2	2	A	**4.32**	7.98	5.15	5.85	11.08	20,20
Plane	1000	11	2	A	20.87	20.55	**18.01**	19.95	27.23	20,20
STAGGER	120	3	2	A	**14.33**	16.08	17.75	20.25	32.52	10,10
FLORA-M	500	6	2	A	**5.34**	10.19	6.32	6.02	16	10,10
FLORA-S	500	6	2	A	**7.7**	12.46	9.21	9	18.9	10,10
COLD	753	128	4	R	**6.04**	7.83	8.23	9.16	35.46	10,30

datasets. For IRIS, CAR and the COLD datasets, the error was averaged over 10 runs on the same dataset.

DACC has the smallest classification error in all cases, except for the CAR and *Plane* datasets. The results on the CAR dataset suggest that a deletion threshold of 0.6 is adapted to the CAR learning problem. Hence, removing learners with a classification accuracy smaller than 60% allows the ensemble to adapt to the simulated concept changes. For the *Plane* dataset, the difference between DACC and CDC 0.7 is likely due to the noise in the *Plane* dataset. Generally, the use of a max function for the global prediction (as in DACC) instead of a weighted combination (as in CDC) affects the predictive accuracy in noisy environments. It results in this case in a higher classification error for DACC by a margin of 2.86%.

6 Conclusion and Future Work

This paper presents an analysis of two main forgetting strategies used by existing online ensemble methods to adapt to concept drifts: (a) deleting experts with poor predictive performance, according to a preset threshold value, and (b) deleting periodically the worst expert in the ensemble. The ensuing analysis lead to the definition of a new approach (DACC) to handle concept drifts.

The analysis shows that the forgetting strategy r(a) equires prior knowledge on the dynamics of the environment in order to choose an adapted threshold value, while strategy (b) may result in unwanted behavior, affecting the ability of the ensemble to adapt to new trends.

DACC deletes periodically one expert chosen randomly from the worst half of the ensemble. According to our study, this strategy corrects unexpected behaviors of the latter forgetting strategy. Empirical comparisons with CDC, a

representative method based on the former forgetting strategy, show that DACC overcomes the difficulty of finding the appropriate threshold, and this on a large variety of concept drifts, with several levels of severity and speed.

For future work, we plan to study another key component of the forgetting strategy: the way experts are weighted and the way their decisions are combined in the ensemble's final decision.

References

1. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 226–235. ACM (2003)
2. Stanley, K.O.: Learning concept drift with a committee of decision trees. Informe Técnico: UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, USA (2003)
3. Tsymbal, A., Pechenizkiy, M., Cunningham, P., Puuronen, S.: Dynamic integration of classifiers for handling concept drift. Information Fusion 9(1), 56–68 (2008)
4. Scholz, M., Klinkenberg, R.: An ensemble classifier for drifting concepts. In: Proceedings of the Second International Workshop on Knowledge Discovery in Data Streams, Porto, Portugal (2005)
5. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., Gavaldà, R.: New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2009)
6. Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: A new ensemble method for tracking concept drift. In: Third IEEE International Conference on Data Mining, ICDM 2003, pp. 123–130 (2003)
7. Karnick, M., Ahiskali, M., Muhlbaier, M.D., Polikar, R.: Learning concept drift in nonstationary environments using an ensemble of classifiers based approach. In: IEEE International Joint Conference on Neural Networks, IJCNN 2008, pp. 3455–3462. IEEE (2008)
8. Minku, L.L., White, A.P., Yao, X.: The impact of diversity on online ensemble learning in the presence of concept drift. IEEE Transactions on Knowledge and Data Engineering 22(5), 730–742 (2010)
9. Schlimmer, J.C., Granger, R.: Beyond incremental processing: Tracking concept drift. In: Proceedings of the Fifth National Conference on Artificial Intelligence, vol. 1 (1986)
10. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning 23(1), 69–101 (1996)
11. UCI Machine Learning Repository, `http://archive.ics.uci.edu/ml/`
12. The COLD Database, `http://www.cas.kth.se/COLD/`
13. Guillaume, H., Dubois, M., Emmanuelle, F., Tarroux, P.: Temporal Bag-of-Words-A Generative Model for Visual Place Recognition using Temporal Integration. In: VISAPP-International Conference on Computer Vision Theory and Applications (2011)

Investigating Facial Features for Identification of Emotions

Giampaolo L. Libralon[1] and Roseli Ap. Francelin Romero[2]

[1] Federal Institute of Sao Paulo (IFSP) - Sao Carlos, SP, Brazil
glibralon@ifsp.edu.br
[2] University of Sao Paulo (USP) - Sao Carlos, SP, Brazil
rafrance@icmc.usp.br

Abstract. The recognition of emotions from others' faces is a universal and fundamental skill for social interaction. Many researchers argue that there is a set of basic emotions which were preserved during evolutive process because they allow the adaption of the organisms behavior to distinct daily situations. In these sense, this paper investigates emotion recognition based on sets of facial expression elements. Different feature sets are proposed to represent the characteristics of the human face and an analysis of the performance of each one is evaluated by Machine Learning techniques. It will be shown that the use of predefined areas of the face in conjunction with angles and distances is a valid proposal to construct models for emotion classification.

Keywords: Emotion Recognition, Facial Expression, Human-Computer Interaction, Machine Learning.

1 Introduction

Social interactions are complex, so each human being presents specific behaviors. In relation to interaction, emotions have a fundamental role. The capacity to express emotions during social interactions is an area with growing importance in the last years. Several mechanisms, which explores distinct ways to express emotion using body, facial expressions, colors, sounds and so on, have been developed in literature [1] [7].

According to Picard [11], for computers to be genuinely intelligent and to interact naturally with humans, they need the ability to recognize, understand and even to have and express emotions. Emotion is fundamental to human experience, influencing cognition, perception, and everyday tasks such as learning, communication, and even rational decision making. Also, they can act as a control and learning mechanism, driving behavior and reflecting how the system is affected by, and adapts to, different factors over time.

Automatic emotion recognition is a multidisciplinary task involving different research fields such as psychology, computer vision, speech analysis and machine learning. Different methods, such as voice intonation, body movements and facial

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 409–416, 2013.

expressions are used by humans to express emotions. Even though body movements or voice are used to express them, emotions are more precisely described by facial expressions, without the need to analyze gestures or voice [4].

Facial expression figures in research on almost every aspect of emotion, including psychophysiology, perception, emotional disorders, and others [1] [7]. This paper focus on the task of emotion recognition based on facial expression analysis of frontal pictures. The six basic emotions presented by Ekman [2], to know: happiness, sadness, anger, surprise, fear and disgust are investigated. Different facial feature sets are proposed and analyzed by Machine Learning (ML) techniques. The evaluated ML techniques are Support Vector Machines (SVMs) and C4.5 algorithm.

2 Facial Analysis for Emotion Recognition

Emotionally intelligent systems must be able to create an affective interaction with users: they must be endowed with the ability to perceive, interpret, express and regulate emotions [11]. Recognize users' emotional state is then one of the main requirements for computers to successfully interact with humans.

In the psychological and cognitive science literature, there are two primary views on the representation of emotions: categorical and continuous. Ekman and others [2] argue for a set of basic emotions, and a set of facial expressions related to them, which are innate and universal across cultures. All other emotional categories are then built up from combinations of these basic emotions. Russell [12] argue that all emotions lie in a continuous two-dimensional space, where the dimensions are typically taken to be valence (how positive or negative the emotion is) and arousal (the energy or excitation level associated with the emotion). Both representations have been used in various computing and robotic applications and the underlying representation does not seem to have a major effect on people's understanding of the machine's emotion.

The task of automatic facial expression analysis can be divided into three main steps: face detection, facial feature extraction and classification. Face detection automatically finds the face region for the input images or sequences. In facial feature extraction, mainly two types of approaches are employed: geometric feature-based methods and appearance-based methods. The geometric facial features present the shape and locations of facial components (including mouth, eyes, eyebrows and nose). The facial components or facial feature points are extracted to form a feature vector that represents the geometry of the face. The appearance facial features present the appearance (skin texture) changes of the face, such as wrinkles and furrows. Both, geometric and appearance features can be also used in a system. Facial expression classification is the last stage. Many classifiers have been applied to emotion recognition such as neural networks (NNs), support vector machines (SVMs), linear discriminant analysis (LDA), K-nearest neighbor, Hidden Markov models (HMM), and others.

2.1 Facial Features Investigated

Six different facial feature sets, which are able to encode distinct aspects of the face: peculiar aspects of the facial expression during each emotion and dynamics or differences of the face according to its neutral state, have been proposed. The vision system employed for facial detection was the Face Tracker [13].

All feature sets proposed in this paper, denoted from now on as FS_1, FS_2, FS_3, FS_4, FS_{1-3} and FS_{2-4}, are based on geometric facial features. To obtain them, Face Tracker was modified to map only 33 feature points, instead of the 66 originally mapped. A small set of original points was chosen based on previous experiments in which the face was mapped only by feature points, considering different subsets of them. Each subset was chosen by selecting a different amount of points to describe the elements (mouth, eyes, eyebrows, chin and nostrils) of the face. So, three distinct subsets were analyzed: the 66 original Face Tracker points, a subset of 33 selected points and a reduced subset with only 26 points. The obtained results were not satisfactory, but they pointed out that mapping the face relying only in feature points is not sufficient to describe a human face for emotion recognition. As the best achieved results were for the 33 points subset, that number of points was adopted for the feature sets presented here, but new facial information was added to improve the facial modeling.

The elements considered in the proposed feature sets aim to model the face parts which are intrinsically related to movements due to emotion expression according to psychologists [2] [3]. Figure 1 presents the graphical elements of the proposed feature sets, illustrated in an image of the Radboud Faces database [6]. As mentioned, they are based on 33 facial points: eight mapping the mouth, six for each one of the eyes, three for each eyebrow and the chin, two for nostrils and two delimiting the lateral extremities of the face near the eyes. The points are represented by red dots. Also, to model the shape of the eyes, the mouth and of face regions related to emotional muscular movements, eight areas are mapped, which can be identified by the geometric regions delimited by the black color line segments.

To obtain FS_1 and FS_2, different distances and angles among the 33 feature points are considered. For the first one, in all possible combinations of points, the distances and the angles that the line connecting two distinct points makes with the horizontal axis are obtained. It creates a feature set of dimensionality $D_1 = 2 \bullet 33 + 8 + 2 \bullet 528 = 1130$.

For the second one, only a subset of the distances and angles from FS_1 are calculated. We chose only the distances and angles which map the regions of the eyes, the mouth, the upper mouth lip with nostrils and the lower mouth lip with the chin. This subset is able to describe mouth and eye states and also movements of the mouth in relation to the chin or nostrils. Considering that the eight mapped areas can correctly represent the remaining emotional facial movements not mapped by the absent angles or distances, this feature set can indeed be representative, with the advantage of the low dimensionality, which is $D_2 = 2 \bullet 33 + 8 + 2 \bullet 107 = 288$. The main goal of FS_2 is to avoid the calculations of all distances and angles from FS_1, which can result in redundancy.

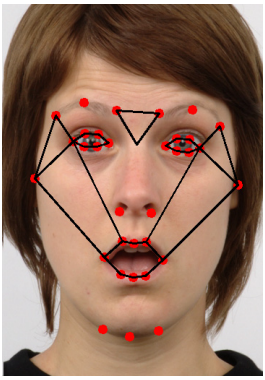


Fig. 1. Graphical representation of the proposed feature sets

Also, the extra information from FS_1 may slow down the ML generalization process and lead to the generation of high complexity and specialized classifiers, resulting in poor performance.

These two feature sets represent only single frame-based features, no information about the relation of these measurements to their values in a frame displaying a neutral expression is encoded. To capture this information, feature sets FS_3 and FS_4, based respectively on the single frame-based feature sets FS_1 and FS_2, were created. These feature sets compare the changes in feature values between the current frame, which represents an emotion, and the frame in which the neutral expression of the subject is present. It is important to highlight they only represent the differences of the values (points coordinates, distances, angles and areas), not the values itself.

The FS_{1-3} and FS_{2-4} are, respectively, a combination of FS_1 with FS_3, and FS_2 with FS_4. The objective is to investigate whether combinations of the created feature sets, which map distinct characteristics of a subjects' expression, can result in better performance for the task of emotion recognition.

3 Experiments and Results

The experiments were performed using the Weka simulator with SVMs or the C4.5 algorithm as classifiers [5]. The 10-fold cross validation methodology [10] was applied for training the classifiers. The proposed system should be able to recognize emotions of any person, including people who were not encountered previously. To report on person-independent system performance, it is important to partition the data in such a way that there are no data from one subject in both the training and the test sets. In this sense, data was partitioned in 80% of samples used for training and the remaining 20% for tests. Samples from a given subject are not present in both training and test sets.

The ML techniques investigated have different parameters to set. To find the optimal parameters in terms of classification performance, it is important that

the whole parameter optimization process is done independently of the test set [14]. To find optimal parameter values, a separate 3-fold cross validation loop [10] is employed each time a classifier is training when searching for the optimal parameters. The classifier performance is evaluated for different values of each parameter. This process is repeated for each of the three folds and the final chosen parameter value is the average over the values found for the three folds. Parameters are assumed independent and there is no specific order in which they are optimized. In all the reported experiments, optimization for all unknown parameters was performed this way.

The data analyzed consists of the Radboud Faces database (RaFD) [6] and the Extended Cohn-Kanade (CK+) database [8]. Both databases are freely available and all displayed facial expressions present information about emotion and were also coded according to the Facial Action Coding System (FACS) [3]. For the performed experiments, only subsets of the RaFD and CK+ databases were used. For RaFD, it contains all images in which models are in frontal view, with eyes directed straight ahead and expressing one of six basic emotions [2]. This resulted in 67 subjects, including both adults and children, with 67 samples from each of the analyzed emotions. For CK+, it possesses information from 50 subjects and contains only image sequences of the same emotions considered for the RaFD database. Also, a balanced number of emotion expressions were kept, resulting in 50 samples from each of the analyzed emotions.

Rotation and translation were applied to each image before their analysis by ML techniques. Rotation aligns the face to the horizontal axis and translation puts it in first quadrant. Also, each image was normalized by its intra-eyes distance. This was done to correct problems of subjects being at distinct distances from the camera, presenting diverse head inclinations or possessing different face formats.

The best results for both data sets can be seen in Table 1. The explored SVM kernels were Linear, Polynomial and Gaussian and all kernel dependent parameters were optimized. For C4.5 algorithm, optimized parameters were the number of minimum instances per node and the Confidence Factor. In general, best SVMs results presented high accuracy rates and were obtained with Gaussian kernels. Accuracy rates for RaFD were from 5% to 19% better than for CK+, depending on the feature set evaluated. The C4.5 algorithm results also achieved mainly high accuracy rates, however not as good as the SVMs ones. Besides, it is important to note the accuracy rates for RaFD were 4% to 18% better than the ones obtained for CK+, depending on the analyzed feature set.

When comparing different approaches, a statistical test is needed to determine the superiority of a particular one among others. Statistical tests were carried out to determine whether any feature set analyzed resulted in better performance, with 95% of certainty, according to *Student t* test [9], for each of the ML techniques investigated.

According to results presented, when using the same classifiers, for RaFD data set, SVMs with FS_{1-3} presented better results, with 95% of certainty,

Table 1. Accuracy obtained by SVMs and C4.5 algorithm for RaFD and CK+ data sets

Feature Set	RaFD data set			CK+ data set		
	SVMs		C4.5	SVMs		C4.5
	Kernel	Accuracy (%)	Accuracy (%)	Kernel	Accuracy (%)	Accuracy (%)
FS_1	Gaussian	80.22 ± 1.05	71.43 ± 1.01	Gaussian	69.24 ± 1.08	58.42 ± 1.03
FS_2	Gaussian	75.82 ± 1.12	71.43 ± 0.97	Gaussian	70.27 ± 1.11	64.87 ± 1.00
FS_3	Linear	90.11 ± 1.07	86.81 ± 1.04	Gaussian	79.49 ± 1.09	68.67 ± 1.06
FS_4	Linear	87.91 ± 1.03	79.12 ± 1.01	Gaussian	79.73 ± 1.06	75.68 ± 1.04
FS_{1-3}	Gaussian	94.51 ± 1.08	84.62 ± 1.03	Gaussian	75.81 ± 1.11	70.97 ± 1.05
FS_{2-4}	Gaussian	90.11 ± 1.09	84.62 ± 1.03	Gaussian	79.17 ± 1.13	79.17 ± 1.04

than with FS_{2-4} and FS_3. However, for C4.5 algorithm, FS_3 presented better results than FS_{2-4} and FS_{1-3}, with 95% of certainty. For CK+ data set, SVMs with FS_4 performed better, without 95% of certainty, than with FS_3 and FS_{2-4}. While FS_{1-3} performed statistically worse. However, for C4.5 algorithm, FS_{2-4} presented better results than FS_4 and FS_{1-3}, with 95% of certainty.

Besides, the statistical analysis also demonstrated, with 95% of certainty, that results for RaFD data set were better. It assures the already proven hypothesis that images with less variability conditions in quality or lighting yields better generalization for ML techniques. Also, SVMs performed better than C4.5 algorithm, with 95% of certainty, for this data set, but this it not true for CK+ data set, in which performance for both ML techniques was quite similar.

As can be noted, for RaFD data set, the best performance was achieved with feature sets FS_{1-3}, FS_{2-4} and FS_3, and the worst can be seen with feature sets FS_1 and FS_2, for SVMs or C4.5 algorithm. For CK+ data set, there is no difference when comparing the worst performance, but the best results for SVMs and C4.5 algorithm were slightly different, even though feature sets FS_4 and FS_{2-4} always appear among best performance. This analysis can lead to an important difference: while RaFD images are of better quality and lightning conditions, resulting in better performance from Face Tracker adopted vision system, the extra information existent in FS_1 provided better generalization capacity for the classifiers investigated. When analyzing the CK+ data set, which possesses images of low quality and worse lighting conditions if compared to the RaFD images, the proposed FS_2, with reduced information to avoid facial data redundancy, consistently improved the final performance for the classifiers investigated. As in real conditions there is no easy way to control image quality or lightning conditions, this feature set seems to be quite adequate, not only because of its reduced dimensionality, but also because of the better performance imposed to the classifiers.

Additionally, the results show the chosen facial characteristics provided to ML techniques the capacity to model the emotion recognition task. It is important to highlight that the feature sets based on facial differences according to its neutral state (FS_3 and FS_4) and the combined feature sets (FS_{1-3} and FS_{2-4}),

which capture differences of the face according to its neutral state as well as characteristics of the facial expression during each emotion, achieved better ML performance, regardless of the data set used. Also, whereas FS_1-based sets have benefited from higher quality images, FS_2-based ones showed themselves robust for less reliable images.

An analysis of which emotion was easily identified by ML techniques was also made. For RaFD data set, the emotions Happiness, Neutral and Disgust were the most easily recognizable. Fear and Anger presented good results, while Surprise and Sadness showed to be the most difficult ones. The only exception is found for Fear when analyzed by the C4.5 algorithm, which presented high error rates, being misclassified mainly as Surprise or Sadness. For CK+ data set, the emotions Happiness, Neutral, Surprise and Disgust were the most easily recognizable. Sadness, Fear and Anger seemed to be the most difficult to identify. One exception can also be found for the C4.5 algorithm, which presented high accuracy rates for emotion Fear, while poor accuracy for Happiness.

It is possible that differences in emotion recognition for each data set have occurred because, for RaFD data set, better results were mainly achieved with FS_1-based sets, while for CK+ data set, best performance was obtained by FS_2-based sets. The Surprise and Sadness recognition problem may be partially explained because Face Tracker presented problems in identifying the mouth wide open, an important characteristic of Surprise, in which was identified the tongue as the lower mouth lip. And also the mouth lips downwards, an intrinsic characteristic of Sadness.

4 Conclusions

The identification of emotions based on facial characteristics was investigated in this paper. For such, geometric features as the shapes of the facial components and the location of facial salient points were used to propose six distinct feature sets which represent characteristics of the human face.

All feature sets were completely developed and validated for the emotion recognition task. An important contribution of this work is in the use of predefined areas of the face in conjunction with angles and distances for constructing feature sets to be used for emotion classification, since the ML techniques presented good performance with high hit rates when applied to both data sets, specially for RaFD data set.

According to FACS [3], distinct facial muscles are activated for each basic emotion. As future work, we intend to investigate the use of the proposed feature sets to analyze the human face in terms of muscular movements or action units (AUs). If these feature sets be able to accurately identify such facial muscles, they can be applied to other areas such as autism, lie detection and son on.

Acknowledgments. The authors thank FAPESP (grant 2008/10554-5).

References

1. Cohn, J.F.: Foundations of human computing: Facial expression and emotion. In: ACM Int. Conf. Multimodal Interfaces, vol. 1, pp. 610–616 (2006)
2. Ekman, P.: Basic emotions. Wiley (1999)
3. Ekman, P., Friesen, W.V., Hager, J.C.: Facial Action Coding System. A Human Face (2002)
4. Ekman, P., Friesen, W.V.: Unmasking the face. Malor Books, Cambridge (2003)
5. Frank, E., Witten, I.H.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann (2005)
6. Langner, O., Dotsch, R., Bijlstra, G., Wigboldus, D.H.J., Hawk, S.T., van Knippenberg, A.: Presentation and validation of the Radboud Faces Database. Cognition and Emotion 24(8), 1377–1388 (2010)
7. Lee, H., Park, J., Chung, M.: A linear affect-expression space model and control points for mascot-type facial robots. IEEE Transactions on Robotics 23(5), 863–873 (2007)
8. Lucey, P., Cohn, J.F., Kanade, T., Saragih, J., Ambadar, Z., Matthews, I.: The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In: IEEE Workshop on CVPR for Human Communicative Behavior Analysis, San Francisco, USA (2010)
9. Mason, R., Gunst, R., Hess, J.: Statistical design and analysis of experiments. John Wiley and Sons (1989)
10. Mitchell, T.: Machine Learning. McGraw Hill (1997)
11. Picard, R.: Affective computing. MIT Press, Boston (1997)
12. Russell, J.: A circumplex model of affect. Journal of Personality and Social Psychology 39, 1161–1178 (1980)
13. Saragih, J., Lucey, S., Cohn, J.: Deformable Model Fitting by Regularized Landmark Mean-Shift. Int. Journal of Computer Vision 91(2), 200–215 (2011)
14. Wessels, L., Reinders, M., Hart, A., Veenman, C., Dai, H., He, T., van't Veer, L.: A protocol for building and evaluating predictors of disease state based on microarray data. Bioinformatics 21(19), 3755–3762 (2005)

Regularly Frequent Patterns Mining from Sensor Data Stream

Md. Mamunur Rashid, Iqbal Gondal, and Joarder Kamruzzaman

Faculty of Information Technology,
Monash University, Melbourne, Australia
{md.rashid,iqbal.gondal,joarder.kamruzzaman}@monash.edu

Abstract. Mining interesting and useful knowledge from the huge amount of data gathered in wireless sensor networks is a challenging task. Works reported in literature use support metric-based sensor association rule which employs the occurrence frequency of patterns as criteria. Such criteria may not be appropriate for finding significant patterns. Moreover, temporal regularity in occurrence behavior should be considered as another important measure for assessing the importance of patterns in WSNs. Frequent sensor patterns that occur after regular intervals is called regularly frequent sensor patterns. Even though mining regularly frequent sensor patterns from sensor data stream is extremely important in many real-time applications, no such algorithm has been proposed yet. In this paper, we propose a novel tree structure called Regularly Frequent Sensor Pattern-tree (RSP-tree) and an efficient mining approach for finding regularly frequent sensor patterns from WSNs. Extensive performance analyses show that our technique is time and memory efficient in finding regularly frequent sensor patterns.

Keywords: Wireless sensor networks, data mining, knowledge discovery, frequent pattern, regularly frequent sensor pattern.

1 Introduction

Recently, wireless sensor networks (WSNs) have emerged as a promising research area with strong practical applications in many areas including environment monitoring, industrial and machine health monitoring, waste water monitoring and military surveillance [1]. A WSN consists of many tiny and low cost heterogeneous or homogeneous sensor nodes that are formed to sense the environment around them and send the detected events to a well-equipped node known as *sink*, through multihop communication. The detected events are transmitted to the *sink* periodically or based upon satisfying a particular predicate or as an answer to a query [2]. In this mode of operation WSNs generate a large amount of data in the form of streams. Such stream data from WSN become useful when they can be mined for knowledge in real time, which presents new challenges for the data mining techniques.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 417–424, 2013.

Recently, sensor association rules have received a great deal of concentration due to their significance in capturing the temporal relationship among sensor nodes in WSNs [7, 8]. An example of sensor association rules could be $(s_1, s_2 \rightarrow s_3, 85\%, \lambda)$ which means that if sensor s_1 and s_2 detect events within time λ, then there is 85% of chance that s_3 detects events within same time interval. of chance that s3 detects events within same time interval.

Another important criterion for identifying the interestingness of frequent patterns might be the shape of occurrence, i.e., whether they occur regularly, irregularly, or mostly in specic time interval in the sensor database. Regularly frequent sensor patterns can be used for predicting the source of future events. By knowing the source of future event, we can detect the faulty nodes easily from the network. For example, we are expecting to get an event from a particular node, and it does not occur signifying that the node might have faulty. Regularly frequent sensor patterns also can identify a set of temporally correlated sensors. This knowledge can be helpful to overcome the undesirable effects (e.g., missed reading) of the unreliable wireless communications.

Traditional frequent pattern mining methods [3, 4] fail to discover such regularly frequent sensor patterns because they only focus on the high frequency pattern. Tanbeer et al. [5] proposed the RP-tree (Regular Pattern tree) to mine the regularly occurs patterns from static transactional databases. Recently, Rashid et al. [6] have introduced a problem of discovering regularly frequent patterns that follow a temporal regularity in their occurrence characteristics. For regularly frequent patterns mining, they proposed a tree structure, called a RF-tree (Regularly Frequent Pattern tree), which capture the database contents in a highly compact manner with two database scans. To find regularly frequent sensor pattern from sensor data stream, we no longer to have the luxury of performing multiple data scans. Therefore, for the two database scans requirement RF-tree is inefficient in mining regularly frequent sensor patterns from the stream of sensor data.

Motivated from the above requirements, in this paper, we develop a single-pass tree structure, called the RSP-tree (regularly frequent sensor patterns tree), that can capture important knowledge from the stream contents of sensor data in a very compact manner. Using a pattern-growth approach, RSP-tree can efficiently mine the regularly frequent sensor patterns from sensor stream. To the best of our knowledge, RSP-tree is the first effort to mine regularly frequent sensor patterns from sensor stream data. Extensive performance study shows that our proposed technique is very efficient in discovering regularly frequent sensor patterns over sensor data stream.

The remainder of this paper is organized as follows. In section 2, we discuss the problem of mining regularly frequent sensor pattern in WSNs. In Section 3, we develop our proposed RSP-tree structure and algorithm. In Section 4, our experimental results are presented and analyzed. Finally, Section 5 concludes the paper.

Table 1. A Sensor database (SD)

TS	Epoch	TS	Epoch	TS	Epoch	TS	Epoch
1	$s_1s_2s_5s_6$	3	$s_1s_2s_5s_6$	5	$s_2s_3s_4$	7	$s_4s_5s_6$
2	$s_1s_2s_3s_5$	4	$s_1s_2s_3s_5$	6	$s_3s_4s_5$	8	$s_2s_3s_4$

2 Regularly Frequent Sensor Patterns Mining Problem in WSNs

Let $S = \{s_1, s_2, ..., s_p\}$ be a set of sensor in a particular wireless sensor network. We assume that the time is divided into equal-sized slots $t = \{t_1, ..., t_q\}$ such that $t_{j+1} - t_j = \lambda, j \in [1, q-1]$ where λ is the size of the each time slot. A set $P = \{s_1, s_2, ..., s_n\} \subseteq S$ is called a pattern of a sensors.

An epoch is a tuple $e(e_{ts}, Y)$ such that Y is a pattern of the event detecting sensors that report events within the same time slot and e_{ts} is the epoch's time slot. A sensor database SD is a set of epochs $E = \{e_1, e_2, ..., e_m\}$ with m = $|SD|$, i.e., total number of epochs in SD. If $X \subseteq Y$, it is said that X occurs in e and denoted as $e_j^X, j \in [1, m]$. Let $E^X = \{e_j^X, ..., e_k^X\}$, where $j \leq k$ and $j, k \in [1, m]$ be be the ordered set of epochs in which pattern X has occurred in SD. Let e_s^X and e_t^X, where $j \leq s < t \leq k$ be the two consecutive epochs in E^X. The number of epochs or time difference between e_t^X and e_s^X, can be defined as a period of X, say p^X. Then a period of X, $p^X = \{e_t^X - e_s^X\}$. Let $P^X = \{p_1^X, ..., p_s^X\}$ be the set of periods for patterns X. For simplicity in period computation, assume the first and last epochs in SD as *null* with $e_f = 0$ and $(e_l = e_m)$ respectively.

Definition 1 (*Regularity of Patten X*): Let for a E^X, P^X be the set of all periods of X i.e., , where n is the total number of periods in P^X. Then the average period value of pattern X represent as, $\bar{X} = \sum_{k=1}^{N} \frac{P_k^X}{n}$ and the variance of periods for pattern X is represent as $\sigma^X = \sum_{k=1}^{N} \frac{\left(P_k^X - \bar{X}\right)^2}{n}$. The regularity of X can be denoted as $Reg(X) = \sigma^X$ (variance of periods for pattern X).

Definition 2 (*Support of a pattern X*): The number of epochs in a SD that contain X is called the support of X in SD and is denoted as $Sup(X) = |E^X|$, where $|E^X|$ is the size of E^X.

Definition 3 (*Regularly frequent sensor Pattern*): A pattern is called a regularly frequent pattern if it satisfies both of the following two conditions: (i) its support is no less than a user-given minimum support threshold, say, min_sup, α and (ii) its regularity is no greater than a user-given maximum regularity threshold say, $max\_variance, \beta$.

Problem definition: Given a SD, $min\_sup(\alpha)$ and $max\_variance(\beta)$ constraints, the objective is to discover the complete set of interesting patterns in SD having than support no less than α and regularity no more β.

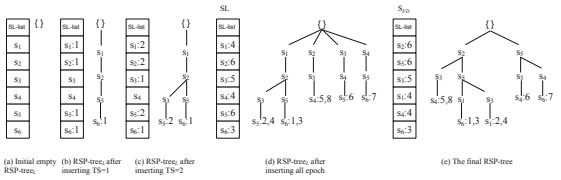


Fig. 1. RSP-tree construction

3 Proposed RSP-Tree Structure and Algorithm

In this section, we present the construction and mining process of the regularly frequent sensor pattern tree (RSP-tree) for finding regularly frequent sensor patterns. The RSP-tree construction has two phases: insertion phase and reconstruction phase. The step-by-step construction process of the RSP-tree is presented below, with examples based on the sensor database of Table I shown in Fig. 1(a-e). For the figure simplicity, we do not show the node traversal pointers in the tree. Each node in a RSP-tree represents a sensor set in the path from the root up to that node. An important feature of a RSP-tree is that, in the tree structure it maintains the appearance information for each epoch. To explicitly track such information, it keeps a list of TS (time-slot) information only at the last sensor-node for an epoch. Such a node is denoted as *tail-node*. Hence, a RSP-tree maintains two types of nodes; namely ordinary node and tail node. The former are types of nodes used in FP-tree that do not maintain TS information. On the other hand, the latter type used in RP-tree [5], can be defined as follows:

Definition 4 (tail node): Let $e = y_1, y_2, ..., y_n$ be an epoch that is sorted according to the SL-list order. If e is inserted into RSP-tree in this order, then the node of the tree that represents item yn is defined as the tail-node for e and it explicitly maintains e's TS. Irrespective of the node type, no node in RSP-tree needs to maintain a support count value like FP-tree. Each node in the RSP-tree maintains parents, children, and node traversal pointers. So, the structures of an ordinary node and a tail node are given as follows: For ordinary node: M, where M is the sensor name of the node. For tail node:$M[e_1, e_2, ..., e_n]$, where M is the sensor name of the node and $e_i, i\epsilon[1, n]$, is an epoch TS in the TS-list, indicating that M is the tail-node for epoch e_i.

Lemma 1: A tail-node in an RSP-tree inherits an ordinary node; but not vice versa.

Proof. The structure of an ordinary node states that it exactly maintains three types of pointers: a parent pointer, a list of child pointers, and a node traversal pointer. A tail-node maintains all such information like an ordinary node. It also maintains the TS-list, which is additional information. Since the TS-list is not maintained in an ordinary node, so we can say, there is an ordinary node in every tail-node and in contrast, no tail-node in an ordinary node.

Insertion Phase: For the insertion phase, RSP-tree arranges the sensors according to lexicographic sensor order in the database and is built by inserting every epoch in database one after another into it and at this stage we call it RSP-$tree_L$. Simply, it maintains a sensor lexicographic order (SL-list). SL-list includes each distinct sensor found in all epochs in database according to sensor lexicographic order (e.g.,$s_1, s_2, s_3, s_4, s_5, s_6$ for example sensor databse show in Table 1) and contains the support value of each sensor in the database. Initially the RSP-tree is empty and starts construction with *null* root node shown in Figure 2(a). First epoch (i.e., TS =1) $\{s_1, s_2, s_5, s_6, s_7\}$ is inserted into the tree $< \{\} \rightarrow s_1 \rightarrow s_2 \rightarrow s_5 \rightarrow s_6 : 1 >$ as-it-is manner and results the first branch of the tree being s_1 as the initial node; just after root node and $s_6 : 1$ is the lail-node is shown in Figure 2(b). Hence, it carries the TS (i.e., 1) epoch in its TS-list. The support count entries for sensors s_1, s_2, s_5 and s_6 are also updated at the same time. Fig. 1(c) shows the status of SL-list and the RSP-$tree_L$ after inserting $TS = 2\{s_1s_2s_3s_5\}$. Ts=2 has its prefix $< \{\} \rightarrow s_1 \rightarrow s_2 >$ common with TS=1. Therefore, the epoch TS=2 is inserted in the tree following the path $< \{\} \rightarrow s_1 \rightarrow s_2 >$ and then creating a new child from s_2 for uncommon part of the epoch with node $s_5 : 2$ being the tail-node that carries the TS information for the epoch.In this way, after adding all epochs (TS=3, TS=4, TS=5, TS=6, TS=7 and TS=8), we get the complete RSP-$tree_L$ shown in Fig. 1(d). We call the SL-list of the constructed RSP-$tree_L$ as SL. Here, the insertion phase is end and the reconstruction phase starts.

Reconstruction Phase: The purpose of the restructuring phase is to achieve a highly compact RSP-tree which will utilize less memory and facilitate a fast mining process. In the restructuring phase, we first sort the SL in frequency-descending order S_{FD} using merge sort and reorganize the tree structure according to S_{FD} order. For restructuring our RSP-tree, we use BSM (branch sorting method) proposed in [9]. BSM uses the merge sort to sort every path of the prefix tree. This approach, first remove the unsorted paths and then sorts the paths and reinserted to the tree. Fig. 1(e) shows the structure of the final RSP-tree that we obtained by restructuring operation.

Property 1: An RSP-tree contains a complete set of frequent sensor projection for each epoch in sensor databse (SD) only once.

Now we describe the mining process of our proposed algorithm. Similar to the FP-growth [4] mining approach, we recursively mine the RSP-tree of decreasing size to generated regularly frequent patterns by creating conditional pattern-bases (PB) and the corresponding conditional trees (CT) without additional database scan. Then we generate the frequent patterns from the conditional tree. At the end we check the regularity of generated frequent pattern to find regularly frequent sensor pattern.

Property 2: The TS-list in a RSP-tree maintains the occurrence information for all the nodes in the path (from that tail-node to the root) at least in the epochs of the list.

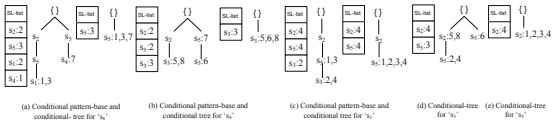


Fig. 2. Conditional pattern-base and conditional tree construction with the RSP-tree

Lemma 2: Let $P = b_1, b_2, ..., b_n$ be a path in a RSP-tree where node bn is the tail node that carries the TS-list of the path. If the TS-list is pushed-up to node b_{n-1}, then the node b_{n-1} maintain the occurrence information of the path $P' = b_1, b_2, ..., b_{n-1}$ for the same set of epochs in TS-list without any loss.

Proof. Based on Property 2, the TS-list at node bn maintains the occurrence information of the path Z' at least in epochs it contains. So, the same TS-list at node b_{n-1} exactly maintains the same epoch information for Z' without any lose.

For example database shown in Table 1, suppose the $min\_sup = 3$ and $max\_var = 1.1$. We explain our mining procedure from below using the bottom-most sensor s_6. The conditional pattern-base tree of s_6 is shown in Fig. 2 (a). According to the Lemma 2, the TS-list of s_6 is pushed-up to its respective parent nodes s_1 and s_4. So, each parent node of s_6 is converted to a tail-node. For node s_6, its immediate frequent pattern is $(s_6 : 1, 3, 7)$ i.e., s6 occurs in epoch 1, 3 and 7 so its support is 3 and it has two paths in RSP-tree: $(s_2, s_5, s_1, s_6 : 1, 3)$ and $(s_5, s_4, s_6 : 7)$ where the number after ":" indicates each sub-pattern occurring TS. Then s_6 conditional pattern-base is $\{(s_2, s_5, s_1 : 1, 3), (s_5, s_4 : 7)\}$ which is shown in Fig. 2 (a). s_6 conditional tree leads to only one branch $(s_5 : 1, 3, 7)$ and the generated frequent patterns are $(s_5 s_6 : 1, 3, 7)$ and $(s_6 : 1, 3, 7)$. We then calculate the regularity of $s_5 s_6$ and s_6 by the Definition 1 andobtain their regularity values 1.5 and 1.5, respectively. Since $\{Reg(s_5 s_6), Reg(s_6)\} > 1.1$, the patterns $s_5 s_6$ and s_6 are not regularly frequent sensor patterns. Similar process is repeated for other sensors in the RSP-tree to find the complete set of regularly frequent sensor patterns which are shown in Fig. 2(b-e).

4 Experimental Results

In this section, we present the experimental results on mining the regularly frequent sensor patterns on the proposed RSP-tree. In absence of any existing method that can mine regularly frequent patterns on sensor data; we compare its performance with the existing method [6] that applies to transactional database. Our programs are written in Microsoft Visual C++ and run with Windows 7 on a 2.66 GHz machine with 4GB of main memory. To evaluate the performance of our proposed approach, we have performed experiments on IBM synthetic dataset (*T10I4D100K*) and real life dataset (*musroom* and *kosarak*) from frequent itemset mining dataset repository [10]. Context and objects in these datasets are similar to the epochs and sensors in the terminology of this paper.

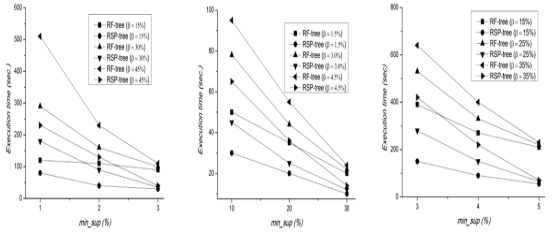


Fig. 3. Execution time comparison

In the first experiment, we show the effectiveness of RSP-tree in mining regularly frequent sensor pattern mining in terms of execution time. To analysis the execution time performance, experiments were conducted with a mining request for the given datasets by varying the *min_sup* and *max_var* values. The x-axis in each graph shows the change of *min_sup* value in the form of percentage of database size and the y-axis indicates the overall execution time. It is shown from the Fig. 3 that, RSP-tree structure outperforms the RF-tree structure in terms of overall execution time in all cases. The reason of this performance gain is that the RF-tree construction requires two database scans, while RSP-tree construction requires only one database scan.

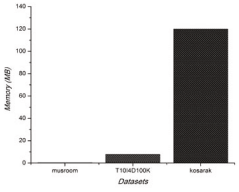


Fig. 4. Compactness of the RSP-tree

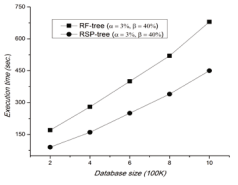


Fig.5. Scalability of RSP-tee

In the second experiment we show the compactness of the RSP-tree. RSP-tree is a threshold independent tree structure. For this reason, we do not compare its memory requirement with RPS-tree. The memory usages of our RSP-tree for different datasets have shown in Fig. 4. Fig. 4 shows that RSP-tree size can easily handle with available memory when it captures the whole database information.

Finally, we study the scalability of the RSP-tree by varying the number of transactions in the database on overall execution time. To test the scalability of RSP-tree, we use kosarak dataset for its huge sparse dataset with a large number of distinct items (41,270) and transactions (990,002). This dataset is divided into five portions each of 0.2 million transactions. The experimental results are present in Fig. 5, where we fix *min_sup*3% and *max_var*40%. From Fig. 5, we can see that as the size of database increases, the execution time increase for RSP-tree and RF-tree, but RSP-tree requires comparatively less execution time with respect of the size of the database.

5 Conclusion

The key contribution of this paper is to provide a novel method for mining regularly frequent sensor patterns over sensor data streams. We have used a pattern growth approach to avoid the level-wise candidate generation-and-test method. The proposed RSP-tree has *build once and mine many* property and is highly suitable for interactive mining. This tree structure require only one database scan to determine the complete set of regularly frequent sensor patterns. Extensive performance analyses show that our tree structure is very efficient for regularly frequent sensor patterns mining and outperform the existing algorithm in both execution time and memory usage.

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine, 102–114 (2002)
2. Boukerche, A., Pazzi, R.W., Araujo, R.B.: A fast and reliable protocol for wireless sensor networks in critical conditions monitoring applications. In: Proc. ACM International Symposium on MSWiM'07, pp. 157–164 (2004)
3. Agrawal, R., Imielinski, T., Swami, A.N.: Mining Association Rules between Sets of Items in large Databases. In: Proc. ACM SIGMOD, pp. 207–216 (1993)
4. Han, J., Pei, J., Yin, Y.: Mining Frequent Pattern without Candidate Generation. In: Proc. of the 2000 ACM SIGMOD, pp. 1–12 (2000)
5. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: RP-Tree: A Tree Structure to Discover Regular Patterns in Transactional Database. In: Fyfe, C., Kim, D., Lee, S.-Y., Yin, H. (eds.) IDEAL 2008. LNCS, vol. 5326, pp. 193–200. Springer, Heidelberg (2008)
6. Rashid, M.M., Karim, M.R., Jeong, B.-S., Choi, H.-J.: Efficient mining regularly frequent patterns in transactional databases. In: Lee, S.-G., Peng, Z., Zhou, X., Moon, Y.-S., Unland, R., Yoo, J. (eds.) DASFAA 2012, Part I. LNCS, vol. 7238, pp. 258–271. Springer, Heidelberg (2012)
7. Boukerche, A., Samarah, S.A.: Novel Algorithm for Mining Association Rules in Wireless Ad-hoc Sensor Networks. IEEE Transactions on Para. & Dist. Sys., 865–877 (2008)
8. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S.: An Efficient Single-Pass Algorithm for Mining Association Rules from Wireless Sensor Networks. IETE Technical Review 26(4) (2009)
9. Tanbeer, S.K., Ahmed, C.F., Jeong, B.-S.: Efficient Single-Pass frequent pattern mining using a prefix-tree. Information Sciences, pp. 559–583 (2009)
10. Frequent itemset mining repository, `http://fimi.cs.helsinki.fi/data/`

Deep Learning Approaches for Link Prediction in Social Network Services

Feng Liu, Bingquan Liu, Chengjie Sun, Ming Liu, and Xiaolong Wang

School of Computer Science and Technology
Harbin Institute of Technology, Harbin, China
{fengliu,liubq,cjsun,mliu,wangxl}@insun.hit.edu.cn

Abstract. With the fast development of online Social Network Services(SNS), social members get large amounts of interactions which can be presented as links with values. The link prediction problem is to estimate the values of unknown links by the known links' information. In this paper, based on deep learning approaches, methods for link prediction are proposed. Firstly, an unsupervised method that can works well with little samples is introduced. Secondly, we propose a feature representation method, and the represented features perform better than original ones for link prediction. Thirdly, based on Restricted Boltzmann Machine (RBM) that present the joint distribution of link samples and their values, we propose a method for link prediction. By the experiments' results, our method can predict links' values with high accuracy for data from SNS websites.

Keywords: Link Prediction, Deep Belief Networks, Restricted Boltzmann Machine.

1 Introduction

Nowdays, a great number of SNS(Social Networking Services) with different interests are available online. Almost all SNS websites allow their social members to interact with each other. The interaction may be showing agreement or disagreement. Taking social members as vertexes in a graph, such interaction can be represented as the 'link' (a direct edge) between them. The interaction of showing agreement can be presented as a link with a positive value between the users, while showing disagreement is a link with a negative value. Because a member's state in the social network is almost valued by these links, estimating the links' values could provide insight into some of the fundamental principles that drive the behaviors of social members.

As defined in [1], link prediction is the problem of predicting the existence of a link between two entities, based on attributes of the objects and other observed links. The predicting task, in this paper, is to predict the attitude of one user toward another from the evidence provided by their relationships with other members of the surrounding social network. Many research about link prediction has been done and several methods have been used.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 425–432, 2013.

In the survey by Linyuan Lü et al. [2], many similarity-based algorithms and probabilistic models for link prediction are introduced. Taskar et al. in [3] and Popescul et al. in [4] use supervised statistical models to predict co-authorship. Brzozowski et al. in [5] use decision tree to predict that weather a user will vote on a resolve under the given conditions. Leskovec et al. in [6] use a logical regression model to predict links' values in signed networks.

In order to improve the performance of statistical models, which are used to solve link prediction, more and more features are taking into account. We did that study in [7]. However, how to improve the performance by representing existing features is seldom discussed. At the same time, all of above methods need a lot of samples and every sample must have a exactly class label. In some conditions, it is not easy to get so many samples with labels. As introduced in [9-11], the high representational ability of DBN based on RBMs suggests us to use deep learning approaches to solve link predicting problems.

In this paper, based on deep learning approaches, we propose an unsurprised method for link prediction, a method to represent features for link prediction and a link prediction method by DBN based on RBMs. The experiments' results show these three methods work well. In section 2, the background knowledge of RBM and DBN are introduced. In section 3, our methods are described. Experiments and results analysis are in section 4 and section 5 is conclusion.

2 Background Knowledge

2.1 Restricted Boltzmann Machine (RBM)

A RBM is a neural network that contains two layers. It has a single layer of hidden units that are not connected with each other. And the hidden units have undirected, symmetrical connections to a layer of visible units. To each unit, including both hidden units and visible units, in the network has a bias. The value of visible units and hidden units are often binary or stochastic units (assume 0 or 1 based on probability).

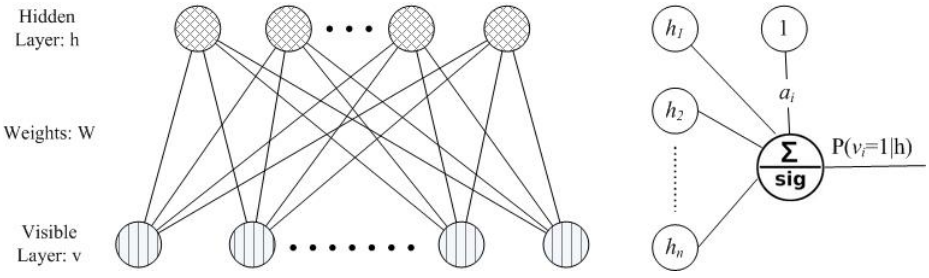


Fig. 1. (a) left is the structure of a RBM (b) right is a visible unit

As shown in figure 1(a), the bottom layer represents a visible vector v and the top layer represents a hidden vector h. The matrix W contains the symmetric, interaction terms between the visible units and the hidden units.

When input a vector v ($v_1,v_2...v_i...$) to the visible layer, the binary state, h_j of each hidden unit is set to 1 with probability by

$$p(h_j = 1 \mid v) = \sigma(b_j + \sum_i v_i w_{ij}) \tag{1}$$

where $\sigma(x) = 1/(1 + e^{-x})$, and b_j is the bias of hidden unit j.

When input a vector h ($h_1,h_2,...h_j...$) to the hidden layer, the binary state, v_i of each visible unit is set to 1 with probability by (as shown in figure 1(b))

$$p(v_i = 1 \mid h) = \sigma(a_i + \sum_j h_j w_{ij}) \tag{2}$$

where a_i is the bias of visible unit i.

RBMs are usually trained by using the Contrastive Divergence(CD) learning procedure, which is described in [8]. To avoid the difficulty in computing the log-likelihood gradient, the CD method approximately follows the gradient of a different function. CD has been applied effectively to various problems, using Gibbs sampling or hybrid Monte Carlo as the transition operator for the Markov chain.

2.2 Deep Belief Network (DBN)

DBNs are multilayer, stochastic generative models that are created by learning a stack of Restricted Boltzmann Machines (RBMs), each of which is trained by using the hidden activities of the previous RBM as its training data. Each time a new RBM is added to the stack, the new DBN has a better lower bound on the log probability of the data than the previous DBN.

One DBN is shown in figure 2(a). Through each layer RBM, the dimension of input visible vector can be decreased, unchanged or increased, when they are represented by the hidden vector. Only the first RBM is trained by the original samples. Then the second RBM is trained by the first RBM's hidden vectors which are generated from the original samples. Do that iteratively until the top RBM is learned. If a sample is inputted to the first RBM of that DBN, the highly abstract vector of that sample would be gotten from the top RBM's hidden layer.

Another DBN is shown in figure 2(b). The network's structure is nearly the same as figure 2(a) except the top RBM. In order to get a model that presents joint distribution of samples and their labels, the labels are transformed to binary vectors firstly. Then the sample label vector is joined with the vector generated by previous RBM from that sample. And get a new vector, which is used to train the top RBM. By such trained DBN, a sample's label can be predicted by trying to join its abstracted vector with all possible label vectors as input for the top RBM.

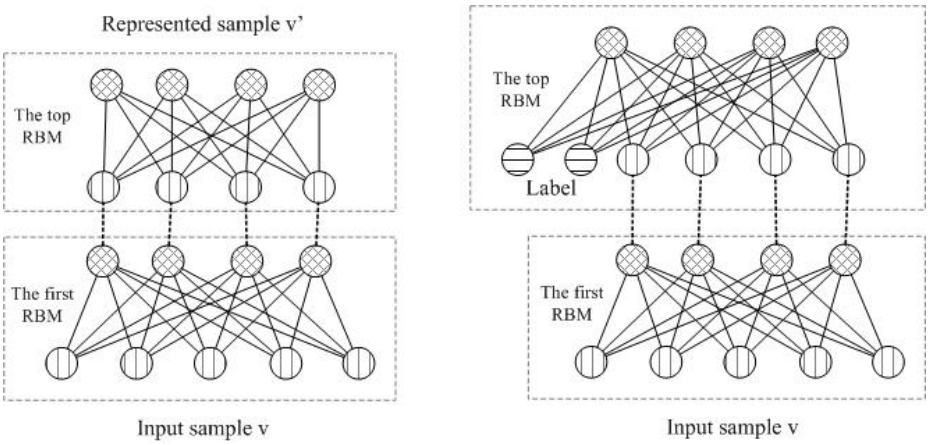


Fig. 2.(a) left is DBN structure for unsupervised link prediction and feature representation. (b)right is DBN structure for link prediction.

3 Deep Learning Approaches for Link Prediction

3.1 Unsupervised Link Prediction

Hinton uses a DBN to learn low-dimensional codes in [10] and shows that work much better than using principal components analysis to reduce the dimensionality of data. With a 4 layer DBN, the image feature's dimension is decreased from 768 to 2. That suggests us to use it in abstracting the features for link prediction

To learn a model for unsupervised link prediction, We use a DBN has the same structure as the one shown in figure 2(a). Through all RBMs, the dimension of input sample vector is decreased. As there are two classes of our link values in our problem, we can predict the input sample's label by the last hidden layer's vector. If the last hidden layer only has one unit, which output is a probability value to be 0 or 1, we can take the probability value as how it look like to belong to a class label. Because the DBN is based on unsupervised training as clustering, the hidden unit's value stands for which label is known. However, the link value can be determined by the most part of the train samples' labels.

3.2 Feature Representation

Well learned RBMs have high representational power. In order to speed up the image retrieval process, only 28 bits are used to represent a 32 x 32 color images in [11]. That suggests us to use RBMs to represent the link prediction features, and use the represented features as another classifier's input.

We use the DBN as shown in figure 2(a) to represent link prediction features. The original feature vectors as the first RBM's visible units' input and use the top RBM's hidden units' output as their represented features. Then we use the represented features to train a logistic regression model to do link prediction.

3.3 DBN for Link Prediction

A DBN is used to recognize handwrite digit images with high accuracy in [9]. The DBN with RBMs forms a very good generative model of the joint distribution of handwritten digit images and their labels. That suggests us to use this method to get the joint distribution of link samples and their values.

We use a DBN as shown in figure 2(b). We transform the link value label to a binary vector. The dimension of the vector is the same as how many classes that we have in samples. At the beginning, set all the bits to '0', then only turn up the *i*th bit to '1', if the sample belong to the *i*th class. We join the class label vector and the hidden value vector form lower RBM as the input for the top RBM. This method could model the joint distribution of link samples and their labels. When test a sample, we use the following metric.

After training the top RBM, each possible label is tried in turn with a test vector and get a set of free energy(by equation (3)) for each combination. Then we can use a Softmax method to get the log probability of which class label should the sample has.

$$F(v) = -\sum_i v_i a_i - \sum_j \log(1 + e^{h_j}) \tag{3}$$

where v_i is the value of visible unit i and a_i is that unit's bias; h_j is calculated by equation (1).

4 Experiments and Analysis

There are three parts of our experiments. The first one is to use a DBN to solve unsupervised link predicting as introduced in 3.1; The second one is to test RBMs' feature representation ability as introduced in 3.2. We get the represented features and test them by a logistic regression model as a classifier; The third one is to use the RBM as a classifier for link prediction as introduced in 3.3.

4.1 Dataset and Features

We use the Wikipedia adminship promotion dataset[1] and features introduced in [7]. In Wikipedia, It allow users to edit others' UGC(User Generate Context) and make commits to them. In order to manage such a huge quantity of UGC, a quite large number of page administers are needed. Wikipedia set up a strategy called Requests for Adminship (RfA) to select administers from normal users.

The features contain the statistics information of users' self degrees and common neighbors. The detail is described in [7], and we get a good prediction result by them. Including the 26 kinds of features, we add another 2 binary features: whether this candidate is supported by a nominator and whether the 'vote' happened in weekend.

[1] http://snap.stanford.edu/

4.2 Experiment Setup

We first normalize all the features to a real value from 0 to 1. The dataset is unbalance for 'support' : 'oppose' as 3.6 : 1, we randomly select a balance subset of 5000 samples from all samples. For the first experiment, different ratio of training and testing sets are used. For the second one, all the 5000 samples are used to get represented features. Then a 5 fold cross validation is performed by a Logistic Regression (LR) model that uses the Newton-Raphson algorithm to calculate maximum likehood estimates. We use 80% for training and 20% for testing in each fold. For the third experiment, we use 50% for training and 50% for testing.

4.3 Results and Analysis

In order to make a comprehensive comparison, we use three evaluation criterions. There are Precision of all samples in test dataset, Area Under Curve (AUC) based on Receiver Operating Characteristic (ROC) and Average Precision (AP). And all Precision values are calculated by the threshold equals to 0.5. In the following result tables, the DBN structure is presented as each RBM with (number of visible units x number of hidden units).

The results of our first experiment are shown in table 1. The unsupervised learning method works well with only 20%(1000) train samples, and gets best result with 80% train samples.

Table 1. Results of unsupervised link prediction method

DBN Structure Two RBMs	**Train: Test**	**AUC**	**AP**	**Precision**
1st (28 x 28) 2nd (28 x 1)	20%:80%	0.722	0.717	67.7%
1st (28 x 28) 2nd (28 x 1)	50%:50%	0.716	0.759	68.6 %
1st (28 x 28) 2nd (28 x 1)	80%:20%	0.820	0.804	71.2%

The results of our second experiment are shown in table 2. The first row of results is got by using the original features as inputs to the LR model. And the following rows are the results of LR with represented features by DBN. By using the represented features, the logistic regression (LR) model's performance is improved about 2%. The result shows that RBM can represent the link prediction features well and the represented features work better for other classifiers, such as a LR model.

Table 2. Results by represented features

DBN Structure Two RBMs	**AUC**	**AP**	**Precision**
Logistic Regression with original features	0.861	0.846	79.7%
1st (28 x 20) 2nd (20 x 20)	0.884	0.899	81.6%
1st (28 x 28) 2nd (28 x 28)	0.886	0.895	81.8%
1st (28 x 56) 2nd (56 x 56)	0.899	0.900	82.1%

The results of our third experiment are shown in table 3. When train the top RBM, we add two label units to the visible layer. We also use the LR model's results used in the second experiment as baseline. Our method's performance is better than the LR model. With adding another RBM to the DBN, the performance is improved.

Table 3. Results of link prediction method based on DBN

DBN Structure Two and Three RBMs	**AUC**	**AP**	**Precision**
Logistic Regression with original features	0.861	0.846	79.7%
Two RBMs 1st (28 x 56) 2nd (58 x 58)	0. 877	0. 957	83.5%
Three RBMs 1st (28 x 56) 2nd (56 x 56) 3rd (58 x 116)	0.891	0. 959	84.6%

By analysis above three experiments' results, our methods work well for link prediction problems. At the same time, we tried to add even more RBMs in the DBN, but the result do not improve as we thought. After analysis on hidden unit values of each layer's RBM, we find the reason is that both the feature's dimension and the labels dimension are low. It does not need so many RBMs to make the features more abstractly. Maybe we will take much more other features into our method in future research, and we have confidence that our method could have better performance.

5 Conclusions

This paper focus on link prediction methods based on deep learning approaches. We see our contribution as follows: (1)We propose a unsupervised learning method for link prediction in social networks. This method has good performance when lack of

training samples, so it can work properly in most conditions. (2) We propose a method to represent link prediction features and these features can improve link prediction performance. This method does not need new features or change the models used in the existing link prediction methods. (3)We tried to use DBN based on RBMs to solve link prediction problem. In the DBN, the top RBM can represent the joint distribution of link values and samples that can predict link values very well.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (61100094, 61272383 and 61300114), and China Postdoctoral Science Foundation (No.2013M530156) .

References

1. Getoor, L., Diehl, C.P.: Link Mining: A Survey. ACM SIGKDD Explorations Newsletter 7(2), 3–12 (2005)
2. Lü, L., Zhou, T.: Link prediction in complex networks: A survey. Physica A: Statistical Mechanics and its Applications 390(6), 1150–1170 (2011)
3. Taskar, B., Wong, M.F., Abbeel, P., Koller, D.: Link Prediction in Relational Data. In: Proceedings of the Neural Information Processing Systems (2003)
4. Popescul, A., Ungar, L.H.: Statistical Relational Learning for Link Prediction. In: Workshop on Learning Statistical Models From Relational Data at the International Joint Conference on Artificial Intelligence, pp. 81–90 (2003)
5. Brzozowski, M.J., Hogg, T., Szabo, G.: Friends and Foes: Ideological social networking. In: Proceedings of the 26th Conference on Human Factors in Computing Systems, pp. 817–820 (2008)
6. Leskovec, J., Huttenlocher, D., Kleinberg, J.: Predicting Positive and Negative Links in Online Social Networks. In: Proceedings of the 19th International Conference on World Wide Web, pp. 641–650 (2010)
7. Liu, F., Liu, B., Wang, X., Liu, M.: Features for link prediction in social networks: A comprehensive study. In: Proceedings of the 2012 IEEE International Conference on Systems Man, and Cybernetics, pp. 1706–1711 (2012)
8. Hinton, G.E.: Training Products of Experts by Minimizing Contrastive Divergence. Neural Computation 14, 1771–1800 (2002)
9. Hinton, G.E., Osindero, S., Teh, Y.: A fast learning algorithm for deep belief nets. Neural Computation 18, 1527–1554 (2006)
10. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
11. Krizhevsky, A., Hinton, G.E.: Using Very Deep Autoencoders for Content-Based Image Retrieval. In: European Symposium on Artificial Neural Networks, Bruges, Belgium (2011)

Spectrum Intensity Ratio and Thresholding Based SSVEP Detection

Akitoshi Itai[1] and Arao Funase[2]

[1] Chubu University, Matsumoto-cho, Kasugai-shi, Aichi 487-8501 Japan
[2] Nagoya Institute of Technology, Showa-ku, Nagoya-shi, Aichi 466-8555 Japan
itai@cs.chubu.ac.jp, funase.arao@nitech.ac.jp

Abstract. Brain Computer Interface (BCI) is a powerful tool to control a computer or machine without body movement. There has been great interest in using Steady-State Visual Evoked Potential (SSVEP) for BCI [1]. Various signal processing and classification techniques are proposed to extract SSVEP from Electroencephalograph (EEG). The feature extraction of SSVEP is developed in the frequency domain regardless of the limitation in hardware architecture, i.e. a low power and simple calculation. We introduced a spectrum intensity ratio as a simple characterization and separation of SSVEP. However, it is difficult to classify an unseeing state of subjects. In addition, we only tried the wide band flickering frequency as visual stimuli. In this paper, we adopt a classification using a simple calculation with threshold to detect the unseeing state from SSVEP in a narrow frequency band.

Keywords: Brain Computer Interface, SSVEP, Spectrun Intensity Ratio, Unseeing Detection.

1 Introduction

BCI provides a direct communication between a human brain and a hardware device [2]. EEG measuring and analysis is one of the major way to transmit a will to BCI. Common characteristics used in BCI are mu and beta wave, Event-Related Potentials and SSVEP [3].

Recent years, SSVEP is often used as a basis for BCI. SSVEP is the periodic EEG response to visual stimuli with a defined or periodic flashing. When we focus our attention or interest on a periodic flickering stimulus, the EEG signal, which includes corresponding frequency and its harmonic of stimuli, is recorded at occipital lobe. The largest response is given at a flickering frequency of 15Hz [4], [5]. SSVEP shows the same fundamental frequency as the flickering visual stimulus and its harmonics [6]. The major SSVEP based BCI adopts the first harmonic [4], first and second harmonics [7], [8], and higher harmonics [9]. Traditional SSVEP detections use an amplitude or a power spectrum to identify the flickering frequency of visual stimuli. M.Cheng used an amplitude spectra of fundamental frequency and second harmonic [10]. The feature extraction and clustering using multi-channel EEG signals achieves a high detection ratio and

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 433–440, 2013.

Table 1. Recording conditions

Stimuli (F)	13 to 18 Hz
Recording length	60 sec.
Sampling rate	1000 Hz
Num. of recording	2

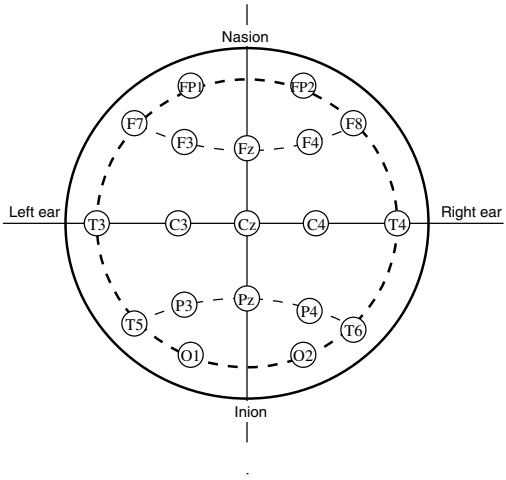


Fig. 1. International 10-20 electrode system

information transfer [11]. If an effective spectrum-based feature extraction is proposed, the simple and reasonable SSVEP detection will be achieved for BCI systems. DFT based feature extraction is employed to extract high-frequency SSVEP [12], however, it requires the baseline spectrum derived from previous EEG data. This means that the subject requires the training time for each trial. We proposed the spectrum intensity ratio (SIR) to extract an enhanced SSVEP [13]. SIR is a ratio of an amplitude spectrum on target frequency to a spectrum around the target frequency, is adopted However, it is difficult to classify an unseeing state of examinees by using our technique. In this paper, we proposed an unseeing state detection based on adaptive threshold.

2 Recording Conditions

Table 1 lists recording conditions for our experiment. The visual stimulus whose flickering frequency is set by a controller is employed to derive SSVEP. The flickering frequency (F) used in our narrow band experiment is 13, 14, 15, 16, 17 and 18 Hz.

The recording is performed in the shielded dark room for reducing an electromagnetic noise and an environmental visual stimulus. The visual stimulus (LED) is located 90cm away from the nasion of the subject. The subject seated in a chair in front of LED looks at a flickering stimulus over 60 seconds. EEG data is collected using 19 electrodes, which are placed at the location based on the international 10-20 system (Fig. 1). EEG is recorded twice. In order to detect an eye movement and blinking, two pairs of electrodes are attached to the right-left side (HEOG) and top-bottom side (VEOG) of a right eye. The reference electrode is placed on both ears.

All potentials are digitally sampled at 1000Hz through the BrainAmp MR Plus (Brain Products Co.) for the off-line signal processing. A high-pass filter

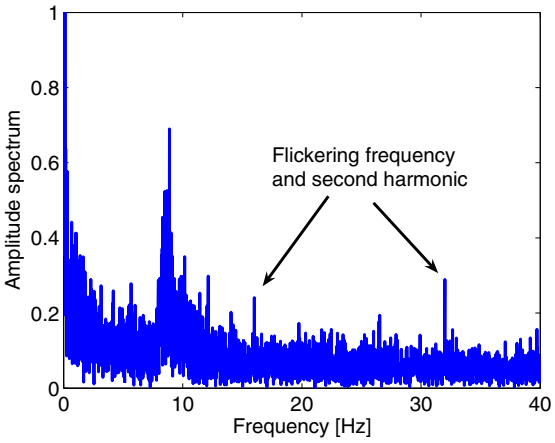


Fig. 2. Amplitude spectrum of EEG recorded for 60 seconds ($F = 16$Hz)

(cut-off 0.53Hz) and a low-pass filter (cut-off 120Hz) is applied to the collected EEG data through the amplifier. The subject is a right-handed male who has a normal vision.

The data selection is important factor to extract SSVEP efficiently. Right occipital lobe shows the better response for flickering visual stimuli [1], [6]. On the other hand, the clear oscillation is given by subtracting EEG recorded at Cz from occipital lobe [14]. We employed a bipolar channel O1-P3 for signal processing [15]. It is well known that frequency peaks of SSVEP appear on the flickering frequency and its harmonics. Fig. 2 draws the amplitude spectrum calculated from 60 seconds EEG when the subject focuses his attention on the flickering stimulus of 16Hz. We can see the sharp spectrum peaks at 16 and 32Hz due to SSVEP. However, EEG strongly includes low-frequency component due to spontaneous activity.

3 Data Analysis

The single-trial analysis using an amplitude spectrum is adopted to detect the fundamental frequency of SSVEP. Assume that $x_i(t)$ is the ith segment of 2 seconds length extracted from EEG. The spectrum of $x_i(t)$ is expressed as $X_i(\omega)$. The sum of amplitude spectrum with overlapping is used to enhance the SSVEP. Then, we get

$$Y_i(\omega) = \sum_{j=0}^{M-1} |X_{i-j}(\omega)| \tag{1}$$

where M is the number of segment for sum.

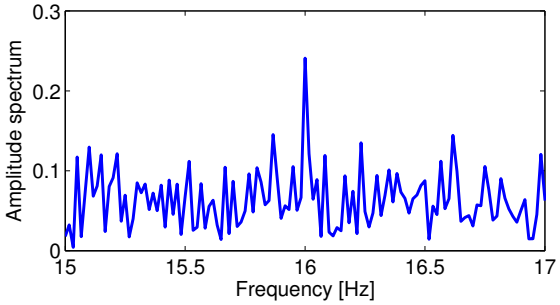


Fig. 3. Amplitude spectrum of 60 seconds EEG (around 16 Hz)

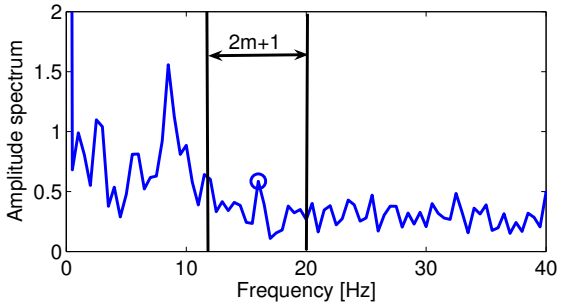


Fig. 4. An example of short term spectrum $Y_i(\omega)$ ($F = 16$Hz)

3.1 Spectrum Intensity Ratio

The SSVEP origin spectrum peak does not appear around the target frequency (see Fig. 3). This is useful characteristics to measure strength of SSVEP from a short term EEG. The spectrum ratio between a recorded EEG and baseline signal is used to enhance SSVEP. However, the normalizing method requires the baseline spectrum calculated from the previous EEG signals [14]. In order to reduce the pre-recording task, the SIR is employed to extract SSVEP. This parameter is represented as a ratio of target frequency to spectrum component around target frequency(see Fig. 4). The SSVEP detection with spectrum intensity ratio is performed as:

$$S_i(\omega_k) = \frac{Y_i(\omega_k)}{\sum_{j=-m}^{m} Y_i(\omega_k + j)}, \tag{2}$$

$$P_i(\omega_k) = S_i(\omega_k) + S_i(2\omega_k), \tag{3}$$

$$\Omega_i = \arg\max_{\omega_k}(P_i(\omega_k)) \tag{4}$$

where ω_k corresponds to the interest frequencies F of LED i.e. 13, 14, ..., 18Hz, Ω_i shows the estimated frequency of SSVEP. The parameter m indicates the bandwidth for feature extraction. We call this detection technique as SSVEP classification.

Table 2. Parameters

Length of DFT window	-	2 seconds
Shift width	-	0.2 seconds
Num. of frames for sum	M	5
Frequency width	m	9 samples

3.2 Unseeing Detection

Unseeing state is that the subject does not focus his attention on the visual stimulus. SSVEP classification does not focus on unseeing state. It is assumed that the subject is always looking at stimulation. We adopt adaptive threshold using average μ and standard deviation σ of SIR to detect unseeing state. The threshold T_i on ith frame is defined as:

$$\mu_i = \frac{1}{L-1} \sum_{\omega_k \neq \Omega_i} P_i(\omega_k), \tag{5}$$

$$\sigma_i = \sum_{\omega_k \neq \Omega_i} \frac{(P_i(\omega_k) - \mu_i)^2}{L-1}, \tag{6}$$

$$T_i = \mu_i + \alpha\sigma_i \tag{7}$$

where, α is a constant number, L is the number of flickering frequency for visual stimuli. To evaluate the unseeing detection technique, we record 60 seconds EEG when the subject does not watch visual stimuli. If the SIR satisfies;

$$P_i(\omega_k) > T_i, \tag{8}$$

$P_i(\omega_k)$ is classified as (4). When (8) is not satisfied, $P_i(\omega_k)$ is detected as unseeing state.

3.3 Detection Ratio

In three methods described above, S_i indicates the estimated frequency of SSVEP which yields the maximum value in each feature extraction. The detection ratio $R_{\omega k}$ is expressed as:

$$R_{\omega_k} = \frac{1}{N} \sum_{i=0}^{N-1} \delta(i) \tag{9}$$

$$\delta(i) = \begin{cases} 1 & (\Omega_i = \hat{\omega}_{ki}) \\ 0 & (\text{otherwise}) \end{cases}$$

where N is the number of frames. The $\hat{\omega}_{ki}$ represents the flickering frequency of the EEG on ith frame. This criterion represents that the how many frames are detected as $\hat{\omega}_k$ and unseeing state. In this paper, the evaluation is performed by using 420 seconds EEG when the subject is looking at 6 stimuli or not looking.

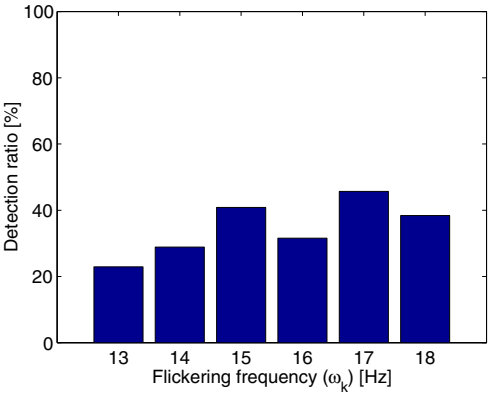


Fig. 5. Detection ratio of SSVEP classification for narrow band stimuli. (1st recording)

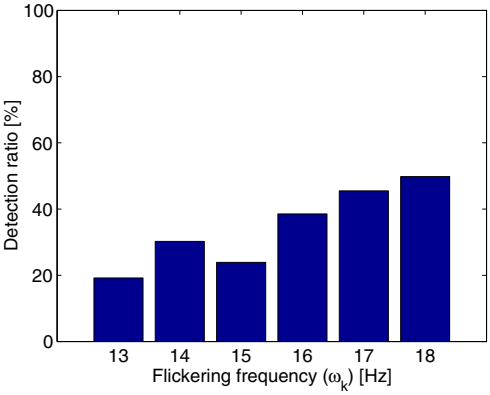


Fig. 6. Detection ratio of SSVEP classification for narrow band stimuli (2nd recording)

4 Experimental Results

4.1 Parameters

Table 2 lists parameters for feature extraction and detection of SSVEP. DFT is applied to EEG recorded for two seconds. The window for DFT shifts at the interval of 0.2 seconds. The number of frame to calculate the sum of amplitude spectrum is 5 related to M in (1). Note that one detection result at ith frame is given by 3 seconds EEG. The bandwidth m for the SIR corresponds to $\omega_k \pm 4.5$ Hz to have a good intensity ratio.

4.2 Detection Results (SSVEP Classification on Narrow Band Stimuli)

At first, we confirm the detection performance of SIR for narrow band visual stimuli. Fig. 5 and 6 draw a detection ratio for 2 recording EEGs. From Fig. 5

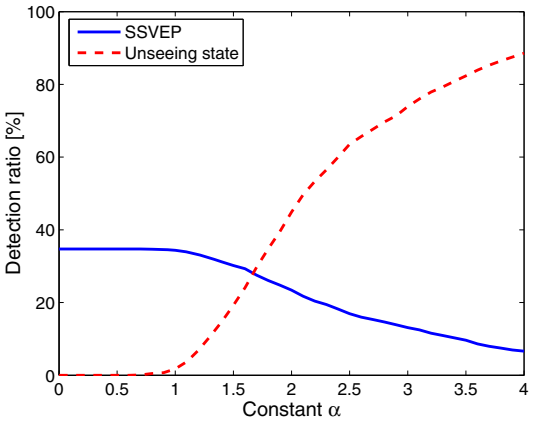


Fig. 7. Detection ratio for SSVEP and unseeing state

and 6, the detection ratio is increasing with a flickering frequency. One possible reason is the increasing of amplitude spectrum at 10 Hz (see Fig. 2). This means that the SSVEP of 13 Hz is not enhanced by the large amplitude spectrum in lower band. On the other hand, the detection ratio of wide-band setting ($F =$ 10, 15, 20 and 25 Hz) is 90% [13] while the SSVEP detection proposed here indicates 35%. From this fact, SIR is useful for the wide-band setting.

4.3 Detection Results (SSVEP and Unseeing State Classification)

The detection ratio of unseeing state depends on the constant value α. Fig. 7 shows the relationship between detection ratio and α. Note that the detection ratio for SSVEP represents the averaged value from 13 to 18 Hz. From this figure, the detection ration of unseeing state is improved rapidly around $\alpha = 1.3$. On the other hand, the ratio for SSVEP is degraded with α. This means that the SSVEP is classified as unseeing state in $\alpha > 1.3$.

5 Conclusion

In this paper, we introduce the spectrum based feature extraction for SSVEP and unseeing state detection. Results show that the detection ratio is 35% for narrow band SSVEP detection. It is confirmed that the SSVEP is not enhanced by using spectrum intensity ratio in the narrow band condition. On the other hand, the unseeing state is detected by using simple threshold. The detection ratio for unseeing state is increasing with constant α while the detection ratio of SSVEP is decreasing. Future task is to unseeing state detection for wide band conditions and define a suitable constant α.

Acknowledgement. This research is supported by gAdaptable and Seamless Technology Transfer Program through target-driven R&D in Feasible Study (FS) stage, Exploratory Research typeh, JST.

References

1. Beverina, F., Palmas, G., Silvoni, S., Piccione, F., Giove, S.: User adaptive BCIs: SSVEP and P300 based interfaces. Psychology Journal 1, 331–354 (2003)
2. Vidal, J.J.: Toward direct brain-computer communication. Annual Review of Biophysics and Bioeng. 2, 157–180 (2003)
3. McFarland, D.J., Miner, L.A., Vaughan, T.M., Wolpaw, J.R.: Mu and beta rhythm topographies during motor imagery and actual movement. Brain Topography 12, 177–186 (2000)
4. Herrmann, C.S.: Human EEG responses to 1-100 Hz flicker: resonance phenomena in visual cortex and their potential correlation to cognitive phenomena. Exp. Brain Res. 137, 346–353 (2001)
5. Garcia, G.: High frequency SSVEPs for BCI applications. Computer-Human Interaction (2008)
6. Regan, D.: Human brain electrophysiology. Evoked potentials and evoked magnetic fields in science and medicine. Elsevier Publisher, NewYork (1989)
7. Cheng, M., Gao, X., Gao, S., Xu, D.: A BCI-based environmental controller for the motion-disabled. IEEE Trans. Neural Syst. Rehabil. Eng. 11, 137–140 (2003)
8. Lalor, E., Kelly, S.P., Finucane, C., Burke, R., McDarby, G.: A brain-computer interface based on the steady-state VEP for immersive gaming control. Biomed. Techn. 49, 63–64 (2004)
9. Muller-Putz, G.R., Scherer, R., Brauneis, C., Pfurtscheller, G.: Steady-state visual evoked potential (SSVEP)-based communication: impact of harmonic frequency components. Journal of Neural Eng. 2, 123–130 (2005)
10. Cheng, M., Gao, X., Gao, S., Xu, D.: Design and implementation of a brain-computer interface with high transfer rates. IEEE Trans. Biomed. Eng. 49, 1181–1186 (2002)
11. Friman, O., Volosyak, I., Graser, A.: Multiple channel detection of steady-state visual evoked potentials for brain-computer interfaces. IEEE Trans. Biomed. Eng. 54, 742–750 (2007)
12. Diez, P.F., Mut, V., Perona, E.A., Leber, E.L.: Asynchronous BCI control using high-frequency SSVEP. Journal of NeuroEng. and Rehabili. 8 (2011)
13. Itai, A., Funase, A.: Spectrum based feature extraction using spectrum intensity ratio for SSVEP detection. In: Annual Int. Conf. of the IEEE EMBS, pp. 3947–3950 (2012)
14. Zhu, D., Bieger, J., Molina, G.G., Aarts, R.M.: A survey of stimulation methods used in SSVEP-based BCIs. Intell. Neuroscience 2010 1, 12 pages (2010)
15. Diez, P.F., Mut, V., Laciar, E., Avila, E.: A comparison of monopolar and bipolar EEG recordings for SSVEP detection. In: Annual Int. Conf. of the IEEE EMBS, pp. 5803–5806 (2010)

Enhanced Hand Shape Identification Using Random Forests

El-Sayed M. El-Alfy

College of Computer Sciences and Engineering,
King Fahd University of Petroleum and Minerals,
Dhahran 31261, Saudi Arabia
alfy@kfupm.edu.sa

Abstract. Over the past ten years, there has been a growing interest in hand-based recognition in biometric technology systems. In this paper, we investigated the application of random decision tree forests for hand identification using geometric hand measurements. We evaluated and compared the performance of the proposed method using out-of-bag validation and 10-fold cross validation in terms of identification. We also studied the impact of the forest size on the performance. The experimental results showed significant improvement over single decision trees, rule-based and nearest-neighbor machine learning algorithms.

Keywords: Biometrics, Hand Recognition, Machine Learning, Decision Trees, Random Forests, Geometric Features.

1 Introduction

Over years biometric technology has attracted the attention of many machine learning, pattern recognition and image processing researchers. It can provide more promising and reliable results to complement or replace traditional personal identification techniques such as keys, passwords, passports and smart cards. Biometric identification uses patterns of one or more intrinsic anatomical or behavioral human traits such as fingerprint, face, iris, voice, gait or signature [1], [2].

One of the lately emerged biometric technologies that has drawn a growing interest is hand-based identification [3], [4], [5], [6]. Its successful deployment can be attributed to several advantageous characteristics. Unlike other biometrics, hand based identification requires less sophisticated, affordable and user-friendly systems. Thus, it can be applied in different environments with low to moderate security requirements. For example, it can be used in work places, airports, and hospitals to access restricted areas and facilities. It can be also used for accessing patient databases in healthcare and medical information systems. Hand image acquisition devices are relatively inexpensive and less complex. In addition, hand geometric features can be easily extracted from low or medium resolution hand images. Moreover, hand based identification is non-intrusive, more convenient and publicly accepted [6]. It also requires low amount of data to identify an

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 441–447, 2013.

individual, has low failure to enroll (FTE) rate, and can be fused easily with other hand-based biometrics (e.g. finger print and veins, palm print and veins, finger and hand shapes, etc.) using same or fewer scanners [7], [8]. Hand geometry has been also used with other biometrics such as face, iris and voice [6].

In a typical system a user should hold his/her hand on a fixed platform with pegs to guide the right position and help feature extraction [9], [10]. However, in more recent work, new techniques with peg-free and/or even contact-less designs are proposed to provide more flexible environments [3], [11], [12], [13].

This paper aims at exploring a new methodology for hand based identification based on random forests of decision trees. This method is evaluated and compared to other machine learning based methods.

The rest of the paper is organized as follows. Section 2 describes distinguishing hand features that can be adopted for hand shape based identification. Section 3 presents the identification method using random forests. The evaluation and discussion of the results are presented in Section 4. Finally, Section 5 concludes the paper.

2 Hand Shape Characteristics

The operation of a hand-based identification system is divided into two phases: enrollment and matching. During enrollment, a number of hand images is captured for each potential user of the system and distinguishing features are extracted and stored in a database. Finally, a computational model for hand based automatic identification is built using a machine learning methodology. During the second phase, the system is used to determine the identity of a person by capturing his hand image and following steps similar to those performed during the enrollment phase to extract distinguishing features. These features are entered to the computational model which outputs the identity that is most likely for the acquired hand image.

Human hands have many distinct features that can be useful in biometric recognition. One of the simplest set of features is the appearance or shape based features [14]. These features often locates important points on the hand shape such as finger tips and valleys and utilizes a limited number of geometric measurements of the hand characteristics in the range of 20 to 30 features. Examples of these features can include lengths and widths of fingers, palm length, hand contour length, etc. After acquiring the hand image, the processing takes place in consecutive steps to segment hand shape from background, remove artifacts, rotate and translate to standard positions, and find important distinguishing features. Figure 1 shows 15 of the features used in this study. These features include 4 features representing lengths of four fingers (little, ring, middle and index) (FL_i), 8 features representing widths of the same four fingers (FW_{ia} and FW_{ib}) at two different locations, palm length and width (PL and PW), and hand length (HL). Two other features, not shown in the Figure, are also used to represent the hand contour length (HCL) and hand area (HA).

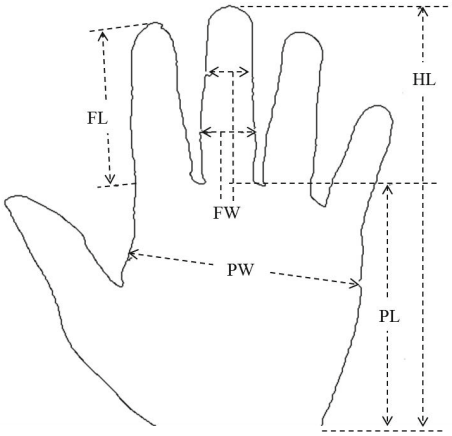


Fig. 1. Typical hand geometry features

3 Identification Method

The decision tree forest consists of an ensemble of decision trees trained independently. The final classification decision is made by aggregating decisions by individual trees using a majority vote rule. Figure 2 shows the typical layout of constructing a decision tree forest. The algorithm for building decision tree forest is random forest and it has been developed by Breiman [15]. Rather than generating trees in series similar to TreeBoost, a decision tree forest builds trees independently in parallel. The overall performance of the forest depends on the performance of individual trees composing the forest and the correlation among them. The upper bound on the generalization error is proportional to the average correlation among trees in the forest. The more correlated the trees, the larger the upper bound on the generalization error. To reduce the correlation among trees, each tree is constructed using a set of instances randomly chosen with replacement from the original training dataset according to a fixed probability distribution.

In contrast to the single decision tree [16], random forest avoids the problem of overfitting noisy training data and thus generalizes better when applied to classify instance unseen during training. Consequently, the construction time for each individual tree in the forest is much less than the time taken to build a single tree model; this can be attributed to not pruning trees in a forest while pruning is necessary to build an optimal single decision tree.

4 Evaluation

In order to evaluate the performance of the proposed methodology, a benchmarking dataset [17] of 1000 left hand images for 100 persons with 10 images

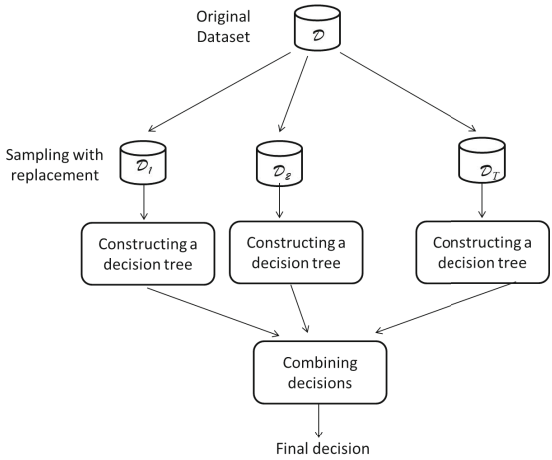


Fig. 2. Typical hand geometry features

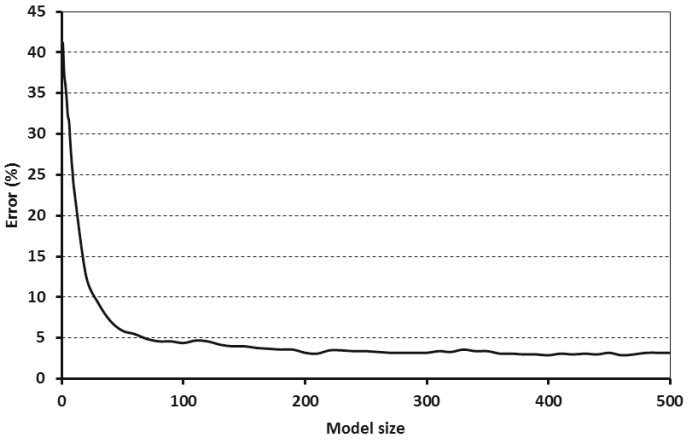


Fig. 3. Tradeoff between error rate and number of trees in the forest

from each person. This dataset has 28 noisy images due to the failure of persons to pose their hands properly. Images are processed and 17 geometric features are extracted for each image as described in Section 2.

In the first experiment, we created and evaluated a decision tree random forest (RF) model. The size of the decision tree forest is controlled by the number of trees in the forest and the size of each tree (depends on the maximum tree depth and the minimum number of instances per node). The results reported in this paper is based on the implementations that exist in the DTREG Software, Enterprise version 9.8.1 [18]. The following parameters were used for the decision tree forest: Maximum trees in the forest was set to 500, maximum splitting levels

Table 1. Identification accuracy comparison of random forest model with other methods

Measure	Model					
	RF	J48	RIPPER	PART	3-NN	5-NN
Accuracy	**96.8**	79.3	64.1	78.5	95.8	94.4

Table 2. Classification error rate for each class when the forest model has 100 trees

Error	Number of Classes	Class IDs
30%	3	61, 79, 81
20%	4	2, 16, 18, 100
10%	15	11, 22, 24, 25, 51, 56, 58, 64, 67, 68, 72, 74, 75, 83, 95
0%	78	all others

was set to 50, minimum size node to split was set to 2, random predictor control was set to square-root of total predictors, and the validation method was set to out-of-bag method. For the sake of comparison, we used single decision tree method (J48), two rule-based methods (RIPPER and PART), and k-nearest neighbor (k-NN) (with $k = 3$ and $k = 5$); these implementations are available in Weka [19].

As depicted by the results in Table 1, the decision tree forest is capable of providing higher identification accuracy than other models. Although nearest neighbor models can provide comparable results, these are lazy classifiers which require storage of the whole dataset and perform comparison during classification; thus no training is conducted. However, other methods build more compact models without requiring the actual instances of the dataset. A disadvantage of decision tree forest as compared to a single decision tree is that it can not be visualized due to the increased complexity. The tradeoff between the percentage error rate and number of trees in the forest is shown in Figure 3. As the number of trees increases, the percentage error decreases and it becomes less than 5% when the number of trees is more than 70. With 100 or more trees, the error becomes less than 3.2%. Table 2 lists the error category for each identity. There are four categories of errors: 30%, 20%, 10% and 0%. Most of the identities are recognized correctly and hence the error is 0%. Figure 4 shows the importance of each feature which is calculated by adding up the improvement in classification gained by each split that used that feature.

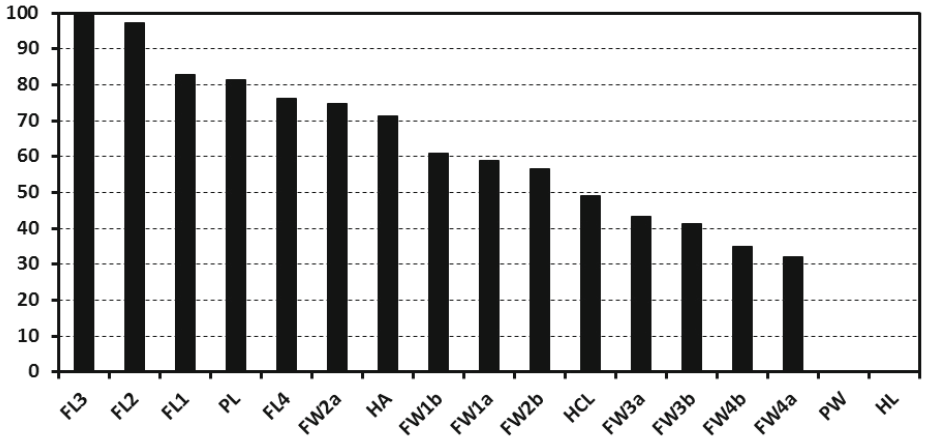


Fig. 4. Importance of each feature when the forest model has 100 trees

5 Conclusion

The performance of decision tree forest for hand shape identification is explored in this paper. The hand shape is described using a limited number of geometric features and an identification model is constructed using the random forest algorithm. The performance is compared with that of single decision trees, rule based, and nearest neighbor methods. The results showed that random forest can be a promising candidate for hand shape identification. Yet it builds relatively complex models as compared to single decision trees. However, in security systems accuracy is a stringent requirement than complexity. In future work, we intend to compare the performance with other methods and using other criteria and datasets.

Acknowledgment. The author would like to thank King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia, and the Intelligent Systems Research Group at KFUPM, for support under grant no. RG1106-1&2.

References

1. Chaudhary, D., Sharma, A.: Hand geometry based recognition system. In: Proc. of 3rd Nirma University International Conference on Engineering (2012)
2. Fong, L.L., Seng, W.C.: A comparison study on hand recognition approaches. In: Proc. of Int. Conf. Soft Computing and Pattern Recognition, pp. 364–368 (2009)
3. Amayeh, G., Bebis, G., Erol, A., Nicolescu, M.: Peg-free hand shape verification using high order zernike moments. In: Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2006)
4. González, S., Travieso, C., Alonso, J., Ferrer, M.: Automatic biometric identification system by hand geometry. In: Proc. IEEE Annual International Carnahan Conference on Security Technology, pp. 281–284 (2003)

5. Sanchez-Reillo, R., Sanchez-Avila, C.: RBF neural networks for hand-based biometric recognition. In: Bigun, J., Smeraldi, F. (eds.) AVBPA 2001. LNCS, vol. 2091, pp. 330–335. Springer, Heidelberg (2001)
6. Nicolae, D.: A survey of biometric technology based on hand shape. Pattern Recognition 42(11), 2797–2806 (2009)
7. Pavesic, N., Ribaric, S., Ribaric, D.: Personal authentication using hand-geometry and palmprint features: the state of the art. In: Proc. of the Workshop on Biometrics (2004)
8. Park, G., Kim, S.: Hand biometric recognition based on fused hand geometry and vascular patterns. Sensors 13(3), 2895–2910 (2013)
9. Sanchez-Reillo, R.: Hand geometry pattern recognition through gaussian mixture modelling. In: Proc. of 15th Int. Conf. Pattern Recognition (2000)
10. Jain, A.K., Duta, N.: Deformable matching of hand shapes for user verification. In: Proc. of Int. Conf. on Image Processing (1999)
11. Morales, A., Ferrer, M.A., Daz, F., Alonso, J.B., Travieso, C.M.: Contact-free hand biometric system for real environments. In: Proc. of the 16th European Signal Processing Conf. (2008)
12. Xin, C., Wu, X., Qiushi, Z., Youbao, T.: A contactless hand shape identification system. In: Proc. of 3rd Int. Conf. on Advanced Computer Control (2011)
13. Guo, J.M., Hsia, C.H., Liu, Y.F., Yu, J.C., Chu, M.H., Le, T.N.: Contact-free hand geometry-based identification system. Expert Systems with Applications 39(14), 11728–11736 (2012)
14. Yörük, E., Dutağaci, H., Sankur, B.: Hand biometrics. Image and Vision Comptuting 24, 483–497 (2006)
15. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
16. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Chapman & Hall, New York (1984)
17. Kumar, A., Wong, D.C.M., Shen, H.C., Jain, A.K.: Personal authentication using hand images. Pattern Recognition Letters 27(13), 1478–1486 (2006)
18. Sherrod, P.H.: Dtreg: Predictive modeling software (2011)
19. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques, 3rd edn. Morgan Kaufmann (2011)

A Novel Application of Particle Swarm Optimisation to Optimal Trade Execution

Marzieh Saeidi and Denise Gorse

Dept of Computer Science, University College London,
Gower Street, London WC1E 6BT, UK
{M.Saeidi,D.Gorse}@cs.ucl.ac.uk

Abstract. Particle swarm optimisation (PSO) is applied for the first time to the problem of optimal trade execution, which aims to partition a large trade so as to minimise hidden costs, here specifically a combination of market impact and opportunity risk. A large order is divided into a set of smaller ones, with both the length of time these remain open and the proportion of the original order they represent being subject to optimisation. It is found that the proposed method can equal the performance of the very popular volume-based VWAP method without in our case having access to trading volume information.

Keywords: Particle swarm optimisation, optimal trade execution.

1 Introduction

With the automation of electronic exchanges the capture of a large volume of market transaction data has become possible and has led to an increased interest in investigating the microstructure of markets in order to discover better trading strategies. An optimal execution strategy is one that maximises opportunities for profit by also minimising trading costs such as *market impact* (the effect one's own actions have on the price of an asset, moving the price up (down) if one's intention is to buy (sell)) and *opportunity risk* (missed opportunity due to a delay in action).

The problem of finding an optimal execution strategy has been approached using a number of different techniques: analytical [1-3]; machine learning [4]; and biologically inspired methods such as genetic algorithms (GAs) [5]. Such methods are useful and especially when the search space is complex, large, or poorly understood. However the current work is to our knowledge the first application of particle swarm optimisation (PSO) to the problem of optimal trade execution.

The data used comprise one year of millisecond microstructure foreign exchange data (EUR/USD), and the aim is to determine a sequence of trades minimising opportunity risk and market impact. It is found that the fitness of discovered solutions compares favourably to naïve strategies such as dividing the order into equal parts held open for identical, non-overlapping time periods, and to the VWAP (volume weighted average price) strategy that is commonly used in the industry.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 448–455, 2013.

2 Financial Background

A well-known problem for traders is that of dividing large orders into smaller ones and submitting them separately so as to minimise market impact. According to Kissell [6] if an order is 15-25% of daily volume it is difficult to execute effectively without being split in this way, while any order higher than 25% of total volume should be split up over several days. In devising an optimal execution strategy there is an inevitable conflict in that trading aggressively will reduce opportunity risk at the cost of increasing market impact, while on the other hand trading too passively will cause a reduction in market impact at the price of increasing opportunity risk. The challenge is to find a balance between these conflicting considerations.

Two kinds of order are utilised in this study. A *market order* is executed immediately but the penalty is that the trader must pay the bid-ask spread, the difference between the highest price a buyer will pay and the lowest price a seller will to sell for. Market orders reduce opportunity risk but if large may have market impact. In the case of a *limit order* buyers and sellers specify the volume and the price they are willing to trade and submit these requests to a *limit order book*, as shown in Figure 1. This has both buy and sell queues, with orders sorted according to price and time priority. A limit order to buy (sell) a certain amount at a price p may be completely or partially executed at a price equal or below (above) p. Limit orders reduce market impact at the risk of incurring opportunity risk. These types of orders together can give a trader a reasonable amount of control over market impact and opportunity risk.

EUR/USD				
	BUY ORDERS		SELL ORDERS	
TIMESTAMP	PRICE	SHARES	PRICE	SHARES
20120102 09:00:00.036	1.29388	100	1.29401	150
20120102 09:00:00.140	1.29388	100	1.29403	50
20120102 09:00:03.071	1.29388	100	1.29402	150
20120102 09:00:03.169	1.29388	100	1.29403	200
20120102 09:00:03.335	1.29388	100	1.29402	150
20120102 09:00:04.866	1.29388	100	1.29403	50
20120102 09:00:04.961	1.29388	100	1.29403	200
20120102 09:00:05.228	1.29388	100	1.29403	50

Fig. 1. Snapshot of a top level of a EUR/USD limit order book (at each timestamp, only the highest price for a buy and the lowest for a sell are displayed)

One further important concept that will be required here is that of a VWAP (volume weighted average price) trading strategy, in which the proportion of an order placed during a particular time period is made proportional to the average volume traded during that period over the past several days. VWAP is a commonly used benchmark; a trading strategy on average executing with a price better than VWAP is considered a good strategy, one achieving a price worse than VWAP the opposite.

3 Particle Swarm Optimisation

There has recently been a growing interest in the application of particle swarm optimisation (PSO) to financial problems [7]. PSO is a stochastic computation technique developed by Kennedy and Eberhart [8] which like GAs is a biologically inspired population-based method but in this case based on a social learning rather than evolutionary metaphor. Studies have shown PSO is at least competitive with GAs, usually achieving equal or better quality solutions at less computational cost [9].

The following is a description of a continuous PSO. A population of i=1..S particles is generated, with each particle's random (subject to satisfying any constraints on the system) initial position representing a solution. At each iteration t each particle i changes its velocity $\mathbf{v}_i$ and position $\mathbf{x}_i$ according to

$$\mathbf{v}_{i,t+1} = w\mathbf{v}_{i,t} + c_1 r_1(\mathbf{p}_{i,t} - \mathbf{x}_{i,t}) + c_2 r_2(\mathbf{g}_t - \mathbf{x}_{i,t}) \quad , \tag{1a}$$

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \mathbf{v}_{i,t+1} \tag{1b}$$

The velocity update rule for a particle is a mix of a push in its previous direction $\mathbf{v}_{i,t}$, of a size governed by the (usually linearly time-decreasing) *inertia weight* w, combined with *cognitive* (following after its own personal best or *pbest* position $\mathbf{p}_{i,t}$) and *social* (following after the swarm's global best or *gbest* position $\mathbf{g}_t$) contributions, in which c_1, c_2 are constant and r_1, r_2 random numbers generated uniformly from [0,1].

4 Methods

4.1 Data

High frequency foreign exchange (FX) data, gathered at millisecond intervals for the currency pairs EUR/USD and USD/CHF, were used for this study. Ideally we would have preferred to use equity data to allow for direct comparison with [5], but due to the limited online resources available to us only access to FX data was possible. However to make our framework more similar to that of [5] the data were filtered to include only equity market hours of 09:00—17:00, giving an 8 hour time horizon for the execution of the order, the size of which was taken to be 10% of daily volume, in the range considered by Kissell feasible within one day "with some work" [7].

The data set, starting from January 2011, was split into 90 day training, 20 day validation, and 20 day test periods, with the last being most chronologically recent. The gbest training strategy doing best on the validation data, denoted *vbest*, is recorded and is the parameter set used for final testing on out-of-sample data. It was noted in 16/20 runs vbest was the same parameter set as the gbest for the last training iteration, evidence overfitting is not here a problem. This is as might be expected given the very large size of the training set (bear in mind these are millisecond data) compared to the size of the search space (16 or 32 dimensional depending on whether only the durations of suborders are allowed to vary (Section 5.1) or volumes also (Section 5.2)).

4.2 Problem Representation

As in [5] the original order is divided into 16 parts, submitted at 30 minute intervals during the day from 09:00 to 16:30. The lifetime of these sub-orders is expressed in minutes and represents the time such an order should stay in the limit order book before being executed at whatever is the then-market price, with the maximum lifetime of an order depending on the time it is submitted, ranging from 480 minutes in the case of the first order at 09:00 down to 30 minutes for the last order at 16:30.

Hence we are initially in Section 5.1 dealing with a 16-dim search space as in [5], with search in each dimension j=0..15 constrained to [0, $\max_j$], where $\max_j$=480-30j minutes. However in Section 5.2 this space will be expanded to include also V_j, the volume traded during each (variable within the appropriate limit) time interval, with the constraint that the sum of the V_j must equal the total volume V of the order.

This representation allows two kinds of extreme strategies to be discovered, one in which all orders are market orders, of lifetime zero, and one in which all orders are defined as limit orders open until the end of the day, when any remaining parts are executed as market orders. While the first of these strategies is little used because traders would thus very strongly signal their intentions, the second, referred to here as 'equal division', is sometimes used, though much less frequently than VWAP.

4.3 Fitness Function

The fitness of a solution (trading strategy) is defined as the average price achieved using this strategy over a given period, averaging over all sub-orders. We note that when executing each sub-order, depending on its size we may need to consume more than one record in the order book. Therefore we calculate fitness as follows

$$F = \frac{1}{S}\sum_{i=1}^{N}\sum_{j=1}^{M_i} p_{i,j} \times s_{i,j} \quad , \tag{2}$$

where N is the number of sub-orders (16 in this case), M_i is the number of records in the order book that was consumed by the ith sub-order, $p_{i,j}$ is the price of the jth record used during this execution, $s_{i,j}$ is the volume of the jth record used, and S is the total volume of the order. Without loss of generality it will be assumed we are deadling with a buy order, so that the objective here will be to minimise this function.

4.4 Learning Algorithm Parameters

In all the experiments to be described below 100 iterations of the PSO algorithm were used, with c_1 and c_2 were set to 2.0 (the most usual choice for these constants), an inertia weight w linearly decreasing from 1.0 to 0.4, and with 20 runs performed under each set of experimental conditions. While some early work used larger swarms, which did not seem to be necessary for good results, the swarm size for the described experiments was set to 27 (= 3^3), reflective of experiments (not here reported) with a Von Neumann local topology as an alternative to a global neighbourhood topology, the intention being to avoid premature convergence of the algorithm. However this

resulted in no significant improvement and was discontinued in favour of a less computationally costly global topology. As will be seen later the bulk of the swarm do not closely follow the leader and it is clear premature convergence does not occur.

4.5 Comparison Strategies

Results from the PSO-based method are compared with four alternative strategies: (i) *market order*, in which a single market order with size equal to the total volume is submitted to the market at the beginning of the day; (ii) *limit order*, in which a single limit order is submitted, with any part of the trade remaining unexecuted at the end of the day being then executed as a market order; (iii) *equal division*, in which the total volume is divided into 16 equal parts with each sub-order, initiated at 09:00, 09:30...16:30, held open as a limit order until the end of the day, when any unexecuted parts are again executed as a market order; and (iv) the VWAP strategy described in Section 2. As noted in Section 4.2 while the first two of these might be considered unrealistic the second two are used in practice, with VWAP being industry standard.

5 Results

Two sets of experiments were performed. In the first set (Section 5.1) the search space was 16-dimensional, the parameters being the durations as limit orders of the equally sized sub-orders places at times of the day from 09:00 to 16:30. In the second (Section 5.2) the search space is expanded to 32 dimensions so as to allow the proportion of the daily trade allocated to each time interval to also be a variable.

5.1 Optimisation of Sub-order Durations with Fixed Volumes

Table 1 shows the average difference, in USD, in the cost of a total daily trade achieved during the 10 week test period of mid-June 2011 to end of August 2011 by PSO and by the comparison strategies, totalled for each day over each millisecond-separated trading opportunity and averaged over 20 separate runs. Recall this was assumed to be a buy trade, so the lower the price achieved the more effective the strategy, a positive value in the table below indicating the relative superiority of PSO.

Table 1. Differences in the overall cost of the trade between the PSO method (Tot_{PSO}) and the four comparison strategies (Tot_{alt}) of Section 4.5

	PSO	Market order	Limit order	Equal division	VWAP
Tot_{alt} - Tot_{PSO}	—	+263.84	+32431.41	+215.05	+196.20

It can be seen that in each case the PSO method is superior, the best of the competitors being the VWAP method, as might be expected given that this is the method most commonly used in practice to divide up a large trade. It is noteworthy that the

proposed method can offer such a challenge to VWAP given that the latter strategy is based on additional information about daily trading volumes. The second best competitor is equal division, also used occasionally. The result that might appear most surprising is that the limit order strategy does much worse than the market order one, in spite of the latter being a poor choice for a large trade as it would very strongly signal a trader's intentions and cause the market to move against him. However this is an effect that could not be replicated here as there was no way to do so without also creating a full market simulator, something outside of the scope of the current work.

The behaviour of the time duration parameters during training was investigated, with the intention of discovering the nature of the solutions reached by the PSO method, and it was found that while on average durations as limit orders converge to around half the maximum allowed for that starting time this is not typical of the fittest, gbest, particle (whose parameters as noted were in most runs also fittest when applied to the validation set). Figure 2 shows the run-averaged gbest durations in the last iteration of training, compared to uniformly decreasing duration weights equal to half the maximum time allowed (from 240 at 09:00 down to 15 minutes at 16:30).

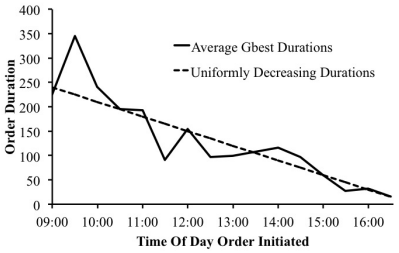


Fig. 2. Run-averaged gbest duration weights compared to uniformly decreasing weights

Limit orders initiated around 09:30 are seen to be left open considerably longer than half of the maximum time then allowed, while conversely sub-orders initiated around 11:30 appear to be handled more quickly. This suggests the PSO algorithm has discovered that some times of the day are more favourable for trading while others are 'dead times' that have to be compensated for by leaving the orders open longer.

One way to investigate this hypothesis is to look at volumes traded at particular times of day, to see if this might be correlated with the pattern displayed in Figure 2. The average EUR/USD trading volume was calculated for equivalent 30 minute intervals during the 90 day training period, with results displayed in Figure 3. It is apparent that activity is peaked between 3:00 and 5:00 pm, with the lowest level of activity around mid-day, and relatively low activity in the morning. Thus it is reasonable that limit orders submitted in the earlier part of the day might need to be left open proportionally longer (compared to the maximum time allowed) so as to wait for better-priced opportunities at a time of day when more liquidity is available.

This trading pattern is close to that behind the volume-based trading strategy VWAP, though it should be re-emphasised our algorithm at this point has no explicit knowledge of market activity, and is not able to apportion different parts of its overall order to different time intervals. This latter will be explored in the next section.

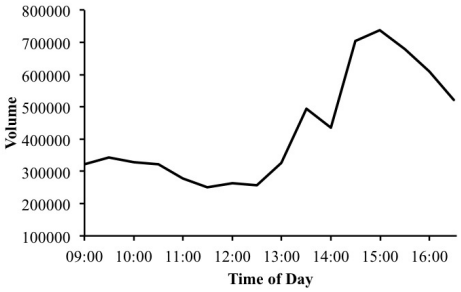


Fig. 3. Average intraday volume for EUR/USD during the training period

5.2 Optimisation of Sub-order Durations and Volumes

Here the search space is extended to 32 dimensions to include the proportion of the overall order handled at each of the 16 daily opportunities, with the length of time each sub-order is open remaining variable as before. Table 2 gives equivalent results to those of Table 1, which show a similar pattern to those of the earlier table.

Table 2. Performance on test data of the PSO algorithm compared to the four alternate strategies, in the case that volumes as well as durations of sub-orders can vary

	PSO	Market order	Limit order	Equal division	VWAP
Tot_{alt} - Tot_{PSO}	—	+274.19	+32441.76	+274.19	+206.55

However the more significant result here is shown in Figure 4, which displays the recommended volume to trade during each 30 minute time slice (09:00-9:30 to 16:30-17:00), accumulated over all sub-orders that extend over the given slice, and where it is seen that the algorithm is recommending that larger amounts be bought at times of higher liquidity. The interesting point is that unlike VWAP the PSO algorithm achieves this result without having prior knowledge about intraday volume.

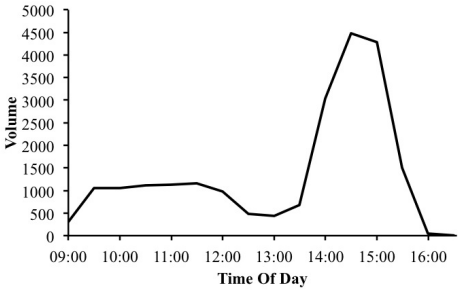


Fig. 4. Recommended volume to trade during each 30 minute time slice

6 Discussion

The trading strategy discovered here by PSO, while demonstrably effective and on a par with VWAP, did not utilise knowledge of recent of recent price histories, or of the trading volumes that are fundamental to VWAP. However the most effective strategy is likely to be one that is responsive to market conditions, trading more or less aggressively as appropriate. By incorporating additional input reflecting recent prices and volumes, and by allowing the starting points of sub-orders to be additional free parameters an aggressive strategy could be achieved by submitting more sub-orders toward the beginning of a time period and a passive one by spreading the orders throughout it. It would also be of interest to seed the initial population with solutions corresponding to the VWAP strategy, and see if in this way the PSO method could be more effectively forced to improve on VWAP. Since results competitive with VWAP have already been achieved it is likely these investigations could result in some substantial improvement over existing popular trading strategies.

References

1. Bertsimas, D., Lo, A.: Optimal Control of Execution Costs. Journal of Financial Markets 1, 1–50 (1998)
2. El-Yaniv, R., Fiat, A., Karp, R., Turin, G.: Optimal Search on One Way Trading Online Algorithms. Algorithmica 30, 101–139 (2001)
3. Almgren, R., Chriss, N.: Optimal Execution of Portfolio Transactions. Risk 3, 5–39 (2000)
4. Nevmyvaka, Y., Feng, Y., Kearns, M.: Reinforcement Learning for Optimized Trade Execution. In: Proc. ICML 2006, New York, pp. 673–680 (2006)
5. Coggins, M., Lim, R.: Optimal Trade Execution: An Evolutionary Approach. In: IEEE Int. Conference on Evolutionary Computation, pp. 1045–1052. IEEE Press, New York (2005)
6. Kissell, R., Glantz, M.: Optimal Trading Strategies: Quantitative Approaches for Managing Market Impact and Trading Risk. Amacom, New York (2003)
7. Banks, A., Vincent, J., Anyakoha, C.: An Review of Particle Swarm Optimization. Part II: Hybridisation, Combinatorial, Multicriteria and Constrained Optimization, and Indicative Applications. Natural Computing 7, 109–124 (2008)
8. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: IEEE International Conference on Neural Networks. IEEE Press, New York (1995)
9. Lee, J.-s., Lee, S., Chang, S., Ahn, B.-H.: A Comparison of GA and PSO for Excess Return Evaluation in Stock Markets. In: Mira, J., Álvarez, J.R. (eds.) IWINAC 2005. LNCS, vol. 3562, pp. 221–230. Springer, Heidelberg (2005)

Stock Price Prediction Based on Hierarchical Structure of Financial Networks

Kanghee Park and Hyunjung Shin*

Department of Industrial Engineering, Ajou University,
Wonchun-dong, Yeongtong-gu, Suwon 443-749, South Korea
{can17,shin}@ajou.ac.kr

Abstract. Stock prices are influenced by many external factors such as the oil prices, the exchange rates, the money interest rates, the certificate of deposit (CD), the gold prices, the exchange rates, the composite indexes in global markets, and so on. And the influence among these factors is reciprocal, cyclic, and often hierarchical, which can be naturally presented as a network. In this paper, a prediction method based on hierarchical structure of financial networks is proposed. Semi-supervised learning (SSL) is employed as a base algorithm, and revised to be suited for time series prediction. A network consists of nodes of the factors and edges of similarities between them. The layered structure of networks is implemented by reforming the existing integration method for multiple graphs. With the hierarchical structure of financial networks, it is able to reflect the complicated influences among the factors to prediction. The proposed method is applied to the stock price prediction from January 2007 to August 2008, using 16 global economic indexes and 200 individual companies listed to KOSPI200.

Keywords: Stock Price Forecasting, Hierarchical Structure, Semi-Supervised Learning (SSL), Machine Learning.

1 Introduction

Stock prince prediction is one of the most difficult issues because of its irregularity of the movement and high uncertainty caused by external economic factors. Many studies on stock price prediction employ various economic factors such as oil prices, exchange rates, interest rates, stock price indexes in other countries, and domestic/global economic situations, etc. [1-4]. Methods for stock price prediction are diverse. Time series analysis is one of the most frequently adopted methods: Jeantheau (2004) predicted stock prices using an ARCH model, and Amilon (2003) and Liu et al. (2009) proposed a prediction method using a GARCH model based on the Skewed-GED Distribution for Chinese stock markets [1-3]. These methods assume that the future data will be varied as similar to those of the past. It is true, and a reasonable result can be obtained from time series analysis if given time series data are

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 456–464, 2013.

originated from natural phenomena, such as the numbers of sunspot, rain-falls, temperature, and so on. However, if data are given from economic or financial factors, it is difficult to expect reasonable prediction performance because of reciprocal and complex influences among the factors. [4]. On the other hand, many studies on the stock price prediction have also been conducted in the machine learning domain. The artificial neural network (ANN) and support vector machine (SVM) methods have been frequently used as a typical model [5-8, 13]. Tay and Cao (2001) proposed a method that introduces financial time series data to the SVM, and Kanas (2003) attempted the prediction of the S&P500 index using the ANN model [9, 10]. Also, Yang et al. (2001) proposed an early warning system of commercial bank loan risks using the ANN model, and Bekiros and Georgoutsos (2008) analyzed that how uncertain news, which show a difficulty in identifying bullish and bearish factors, affect the NASDAQ index using the ANN model [11, 12]. The methods using ANN and SVM may include the interrelation between the stock price and these factors in modeling. However it is still insufficient to explicitly formalize the mutual, complicated, and often hierarchical between the factors.

In this study, we propose a method of stock price prediction by using the hierarchical structure of layers of differently scaled networks: a layer of global economic indicators and a layer of individual stock prices. The former includes global indicators such as certificates of deposit (CD), gold prices, exchange rates, and oil prices, and global composite stock indexes of representative stock exchange markets like NASDAQ, DOW, S&P500, etc. The latter, on the other hand, is composed of individual stock prices of domestic companies, e.g., HYUNDAI Motors, SAMSUNG Electronics, LG Chemicals, and so on. To implement the idea, semi-supervised learning (SSL) is employed as the base algorithm which is most recently emerged a category of machine learning algorithms [14, 17]. With SSL, the reciprocal and complicated relations between economic factors can be naturally implemented. To adapt it for time series prediction, the original formula of SSL are revised and reformulated [15, 16]. To implement hierarchical relations between layers, the SSL based graph-integration method is employed [18, 19].

The rest of this paper is organized as follows. Section 2 briefly introduces the SSL algorithm. Section 3 presents the proposed method: the revised SSL for time series prediction and hierarchical structure. Section 4 provides the experimental results as evaluated with the stock prices of 200 individual companies listed to KOSPI from January 2007 to August 2008. Finally, in Section 5, conclusions will be drawn.

2 Semi-Supervised Learning

In graph-based SSL algorithm, a data point (or entity) $x_i \in R^M$ (i = 1,…, n) is represented as a node i in a graph (or network), and the relationship between data points is represented by an edge where the connection strength from node j to other node i is encoded as w_{ij} of a similarity matrix W [20]. Figure 1 presents a graphical representation of SSL.

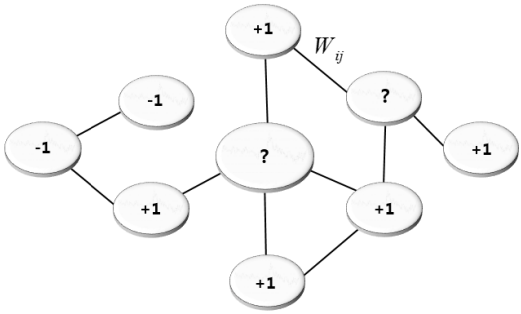


Fig. 1. Semi-supervised learning (SSL): The labeled node is denoted as "+1" or "-1", and the unlabeled node as "?"

A weight w_{ij} can take a binary value (0 or 1) in the simplest case. Often, a Gaussian function of Euclidean distance between points with length scale σ is used to specify connection strength:

$$w_{ij} = \begin{cases} \exp\left(-\frac{(x_i - x_j)^T (x_i - x_j)}{\sigma^2}\right) & \text{if } i \sim j \ (\text{'k' nearest neighbors}) \\ 0 & \text{otherwise} \end{cases}, \quad (1)$$

Usually, an edge $i \sim j$ is established when node i is one of k-nearest neighbors of node j or node i is within a certain Euclidean distance r, $\|x_i - x_j\| < r$. The labeled nodes have labels $y_l \in \{-1,1\} (l = 1, \ldots, L)$, while the unlabeled nodes have zeros $y_u = 0$ (u=L+1,...,L+U). The algorithm will output an n-dimensional real-valued vector $f = [f_l^T \ f_u^T]^T = (f_1, \ldots, f_L, f_{L+1}, \ldots, f_{L+U})^T$ which can be thresholded to make label predictions on $f_{L+1}, \ldots, f_{L+U}$ after learning. It is assumed that (a) f_i should be close to the given label y_i in labeled nodes and (b) overall, f_i should not be too different from its adjacent nodes f_j. One can obtain f by minimizing the following quadratic functional:

$$\mathrm{Min}_f \ (f - y)^T (f - y) + \mu f^T L f \ , \quad (2)$$

where $y = (y_1, \ldots, y_l, 0, \ldots, 0)^T$, and the matrix L, called the graph Laplacian, is defined as $L = D - W$, $D = \mathrm{diag}(d_i)$, and $d_i = \sum_j \omega_{ij}$. The first term corresponds to the loss function in terms of condition (a), and the second term represents the smoothness of the predicted outputs in terms of condition (b). The parameter μ represents trades between loss and smoothness. The solution to (2) is obtained as

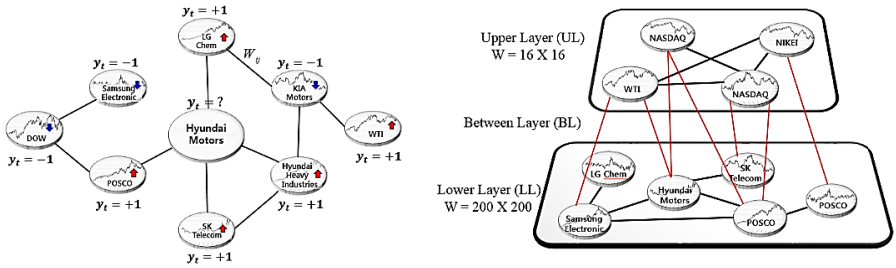
$$f = (I + \mu L)^{-1} y \ , \quad (3)$$

where I is the identity matrix.

3 Proposed Method

3.1 Revised SSL for Time Series Prediction

Assume that a set of time series data are given for the stock price prediction of Hyundai Motors, i.e., the stock prices of individual companies such as LG Chemical and



(a) Network of time series variables. (b) Hierarchical Network.

Fig. 2. The Proposed Method

KIA Motors, WTI intermediate oil price, and other external factors, etc. To apply SSL to this problem, the setup for the original SSL should be revised.

The nodes in the graph represent the factors that influence the stock price of Hyundai Motors, which are all time series variables. Then the edge between any two nodes i~j stands for the similarity of the two sets of time series, represented as ' $w_{ij} \in W$ ' . Auto-correlation of the two time series can be used as a value of similarity. The label 'y_t' in Figure 2(a) implies whether the predicted value of the corresponding variable is up or down at time point t. It can be formulated as follows:

$$y_t = \text{sign}\left(x_t - MA_5(x_t)\right) \quad . \tag{4}$$

For instance, if the stock price of Hyundai Motors exceeds its five-days moving average, (4) will give a '$y_t = +1$' label. On the contrary, the node is labeled as '$y_t = -1$' for the opposite case. And '$y_t = 0$' if there is no information about the movement of the corresponding time-series value at time point t; the label is to be predicted. In Figure 2(a), the label of Hyundai Motors is not known yet at time point t, and hence unlabeled. Given W_{t-1} and Y_t, the up/down movement directions of stock price f_t can be obtained from the formulation of (2) and its closed-form solution (3) of the SSL classification framework.

3.2 Prediction from Hierarchical Structure

The proposed method uses the hierarchical structure which is layered with differently scaled networks. The network in the 'upper layer (UL)' is composed of global economic indicators such as certificates of deposit, gold prices, exchange rates, and oil prices, and global composite stock indexes of representative stock exchange markets like NASDAQ, DOW, S&P500, etc. On the other hand, the network in the 'lower layer (LL)' is composed of individual stock prices of domestic companies, e.g., Hyundai Motors, Samsung Electronics, LG Chemicals, and so on. And there is a latent layer, denoted as 'between layers (BL)', which consists of connections of two factors, one belongs to UL whereas the other belongs to LL, and vice versa. Figure 2 (b) illustrates the hierarchical structure.

To incorporate the hierarchical structure into SSL, the integration method for multiple networks is employed [18, 19]. However, the existing integration method is not

for the 'hierarchical' structure but for 'parallel'. Therefore, in order for utilizing the integration method, it requires an idea of parallel representation for the layers in the hierarchical structure. Figure 3 describes the idea. For each of the layers, an extended similarity matrix is prepared: the extension is performed in order to set the dimensions of multiple matrices are equal, but most elements of which are null-valued. In the figure, W_{UL} represents the similarity matrix in the upper layer, and W_{LL} for the lower layer, and W_{BL} for the between layer. Then, by applying the integration method, we can obtain integrated results of the layers in the hierarchical structure. According to [18,19], integrating multiple networks literally translates to finding an optimum value of the linear combination coefficients α_k's for the individual k graphs. The value of the coefficient tells significance of the corresponding network for prediction.

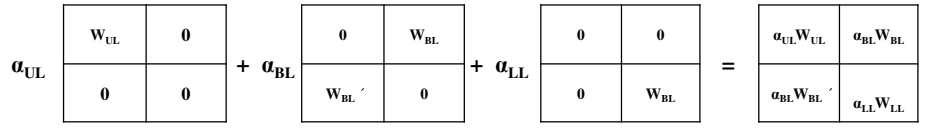


Fig. 3. An idea of parallel representation for the layers in the hierarchical structure

The formulation for integration of the layers in the hierarchical structure is summarized as follows:

$$\mathrm{Min}_{\alpha,f} \ (\mathrm{f}-\mathrm{y})^{\mathrm{T}}(\mathrm{f}-\mathrm{y}) + \sum_{k\in\{LL,BL,UL\}} \alpha_k \, \mathrm{f}^{\mathrm{T}} \mathrm{L}_k \mathrm{f}, \tag{5}$$

$$\mathrm{f} = \left(\mathrm{I} + \sum_{k\in\{LL,BL,UL\}} \alpha_k \, L_k\right)^{-1} \mathrm{y}, \tag{6}$$

where k is the number of layers, and L_k means the graph-Laplacian in the kth layer.

4 Experiment

4.1 Experimental Setting

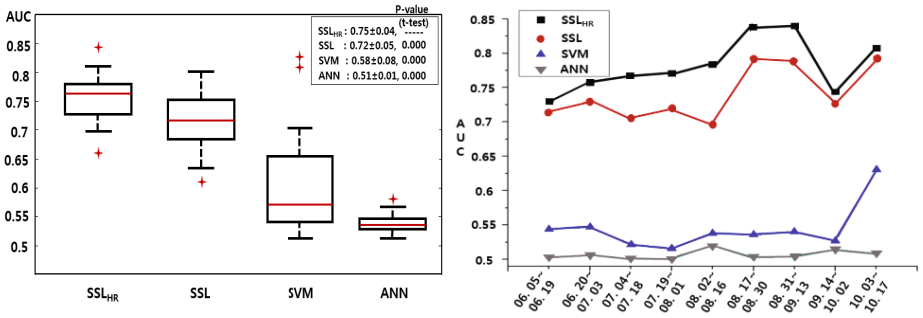
A total of 403 daily prices of 216 economic/financial indicators were used for experiment during the period from January 2007 to August 2008. The 16 global economic indicators which consist of the upper layer of the hierarchical network included Dow-Jones average (DOW), National association of securities dealers automated quotations (NASDAQ), Japanese stock market index (NIKKEI), Hang seng index (HSI), Shanghai composite index (SSE), Taiwan stock exchange corporation (TSEC), Financial times security exchange (FTSE), Deutscher aktien index (DAX), Continuous assisted quotation index (CAC), Bombay stock exchange portmanteau of sensitive and index (BSE_SENSEX), Indice bovespa (IBOVESPA), Australia all ordinaries index (AORD), Korea composite stock price index (KOSPI), Exchange rate(KRW-USD), the west Texas intermediate oil price (WTI), and the certificate of deposit (CD). And the lower layer was composed of the 200 individual stock prices of domestic companies, which are listed to KOSPI200 during the period.

The proposed method--the SSL based on the hierarchical structure of networks (SSL_{HR}), was compared with three representative machine learning models: an ordinary

SSL, an ANN with 3-layered MLP, and an SVM with Gaussian kernel. For each of the four models, the best performance was selected by searching over the respective model-parameter spaces: the optimal number of the hidden node in ANN was determined in the range of {3~50}[21] and the parameters of kernel width (gamma) and misclassification tradeoff (C) in SVM were determined in the range of $\{\text{gamma}, C\} \in \{0.01, \ldots, 100\} \times \{0.01, \ldots, 100\}$ [15]. For SSL_{HR} and SSL, the parameters in (1) and (2) were determined in the range of $\{k, \mu\} \in \{3, \ldots, 50\} \times \{0.01, \ldots, 100\}$. In SSL_{HR}, for calculation of similarities between indicators, the closing/opening/high/low prices and the trading volume were used for the upper layer. For the lower layer, the trading value, PER, the dividend yield, the capital stock, and the listed shares were additionally included for the calculation. And for the 'between-layers' that connects the upper and lower layers, the closing price and trading volume were used which are common in both layers. During the period, the first 103 daily prices were used for training and validation. And the remaining 300 daily prices were reserved for out-of-sample (test) evaluation [22]. The area under the receiver operating characteristic (ROC) curve (AUC) was used as a performance measure [23].

4.2 Results

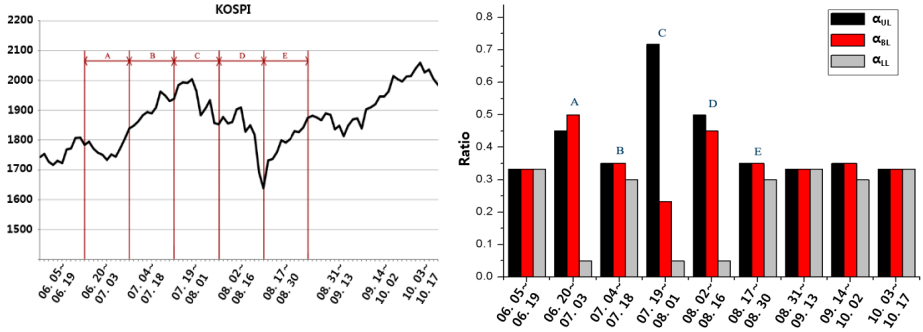
Figure 4 shows boxplots of AUC during the test period. The average AUC value of SSL_{HR} was 0.75(±0.04), which was ranked as the best performance compared with the others: the AUC values of SSL, SVM and ANN were 0.75(±0.04), 0.72(±0.05), 0.58(±0.08) and 0.51(±0.01), respectively. The difference in performance was verified via the pairwise t-test between SSL_{HR} and others, and it showed statistical significance as shown in the upper right panel in Figure 4(a). Figure 4(b) presents the piecewise AUCs measured at two-week intervals during the test period. For every interval, SSL_{HR} showed outperformance than other ones. The second best performance was achieved by SSL. The reason for outperformance of the two models comes from the adaptable structure of SSL to changes in relations between financial-economic factors. During the test period, the relations might have varied, and whenever it varied the similarity matrix of SSL reflected it. On the other hand, both ANN and SVM have difficulties to address the varying relations due to its innate model structure.



(a) Overall average comparison (b) Piecewise comparison

Fig. 4. The performance comparison (AUC): SSL_{HR}, SSL, SVM, and ANN

To elucidate the reason of superiority of SSL_{HR} to an ordinary SSL, the changes in values of the combining coefficient α_k were further investigated, which will reveal the effect of the hierarchical structure of financial networks. On each of the two week intervals marked in the flows of KOSPI, the values of the upper/lower/between layers were measured. See (a) and (b) of Figure 5. For the intervals which shows a pattern of normal steady ups and downs, the values of α_k for each layer showed similar ratios: the intervals A/B/E including the rest period which was not particularly marked. However, for the intervals C/D, the values of α_{UL} and α_{BL} had larger ratios, relatively to α_{LL}. The C/D intervals correspond to the period of the U.S. subprime mortgage crisis that led to a global financial crisis and subsequent recession afterwards. In particular, the value of α_{UL} sharply increased in the intervals C. This can be conjectured that in the interval C which was the pre-period before the crisis, the symptoms for the global crisis might have prevailed among countries. And the precaution might have reflected to global financial indicators, and thus been captured by the upper layer of the proposed hierarchical networks. This implies that the proposed hierarchical SSL model showed robustness against global economic shocks, and can be a good candidate for early warning system.



(a) The five intervals. (b) The ratios in values of α_k's.

Fig. 5. The values of combining coefficients for the upper/lower/between layers

5 Conclusion

In this study, we proposed a prediction method for stock prices based on hierarchical structure of financial networks. To implement this, SSL was employed. We first designed a network consisting of nodes of financial/economic factors and edges of similarities between them. Then, the SSL formula was revised to be suited for time series prediction. The hierarchical structure was composed of differently scaled networks: one consisted of global economic indicators, and the other consisted of individual stock prices of domestic companies. The prediction method from the hierarchical networks was implemented by reforming the existing integration method for multiple graphs [18, 19].

The structural design of the proposed method leads to several benefits. First, it reflects reciprocal, cyclic, and hierarchical influences among the financial factors to prediction, and thus better performance can be obtained. If combined with a portfolio optimization method, better profits and stabilities in investments can be expected. Second, it is robust against global economic shocks, which has a potential to be used as an early warning system.

Acknowledgements. The authors would like to gratefully acknowledge support by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2013R1A1A3010440/2010-0028631).

References

1. Jeantheau, T.: A link between complete models with stochastic volatility and ARCH models. Finance Stochast. 8, 111–131 (2004)
2. Liu, H.C., Lee, Y.-H., Lee, M.-C.: Forecasting China Stock Markets Volatility via GARCH Models Under Skewed-GED Distribution. Journal of Money, Investment and Banking, 5–14 (2009)
3. Amilon, H.: GARCH estimation and discrete stock prices: an application to low-priced Australian stocks. Economics Letters 81, 215–222 (2003)
4. Chen, N.F., Roll, R., Ross, S.A.: Economic Forces and the Stock Market. Journal of Business 59, 383–403 (1986)
5. Kim, K.J.: Financial time series forecasting using support vector machines. Neurocomputing 55, 307–319 (2003)
6. Huang, W., Nakamori, Y., Wang, S.-Y.: Forecasting stock market movement direction with support vector machine. Computers & Operations Research 32, 2513–2522 (2005)
7. Cao, Q., Leggio, K.B., Schniederjans, M.J.: A comparison between Fama and French's model and artificial neural networks in predicting the Chinese stock market. Computers & Operations Research 32, 2499–2512 (2005)
8. Chen, A.S., Leung, M.T., Daouk, H.: Application of neural networks to an emerging financial market: forecasting and trading the Taiwan Stock Index. Computers & Operations Research 30, 901–923 (2003)
9. Tay, F.E.H., Cao, L.: Application of support vector machines in financial time series forecasting. Omega 29, 309–317 (2001)
10. Kanas, A.: Non-linear forecasts of stock returns. Journal of Forecasting 22, 299–315 (2003)
11. Yang, B., Li, L.X., Xu, J.: An early warning system for loan risk assessment using artificial neural networks. Knowledge-Based Systems 14, 303–306 (2001)
12. Bekiros, S., Georgoutsos, D.: Direction-of-Change Forecasting using a Volatility-Based Recurrent Neural Network. Journal of Forecasting 27, 407–417 (2008)
13. Tsang, P.M., Kwok, P., Choy, S.O., Kwan, R., Ng, S.C., Mak, J., Tsang, J., Koong, K., Wong, T.-L.: Design and implementation of NN5 for Hong Kong stock price forecasting. Engineering Applications of Artificial Intelligence 20, 453–461 (2007)
14. Zhu, X.: Semi-Supervised Learning with Graphs, PA 15213. Carnegie Mellon, Pittsburgh (2005)
15. Park, K., Shin, H.: Stock Price Prediction based on a Complex Interrelation Network of Economic Factors. Engineering Applications of Artificial Intelligence 26, 1550–1561 (2013)

16. Shin, H., Hou, T., Park, K., Park, C.-K., Choi, S.: Prediction of movement direction in crude oil prices based on semi-supervised learning. Decision Support Systems 55, 348–358 (2013)
17. Shin, H., Hill, N.J., Lisewski, A.M., Park, J.-S.: Graph sharpening. Expert Systems with Applications 37, 7870–7879 (2010)
18. Shin, H., Tsuda, K.: Prediction of Protein Function from Networks. In: Chapelle, O., et al. (eds.) Semi-Supervised Learning, pp. 339–352. MIT Press (2006)
19. Shin, H., Lisewski, A.M., Lichtarge, O.: Graph sharpening plus graph integration: a synergy that improves protein functional classification. Bioinformatics 23, 3217–3224 (2007)
20. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with Local and Global Consistency. In: Advances in Neural Information Processing Systems 16 (NIPS), Whistler, Britishi Columbia, pp. 321–328 (2004)
21. Kim, K.J.: Artificial neural networks with evolutionary instance selection for financial forecasting. Expert Systems with Applications 30, 519–526 (2006)
22. Liu, Q., Sung, A.H., Chen, Z., Liu, J., Huang, X., Deng, Y.: Feature Selection and Classification of MAQC-II Breast Cancer and Multiple Myeloma Microarray Gene Expression Data. MAQC-II Gene Expression 4, 1–24 (2009)
23. Gribskov, M., Robinson, N.L.: Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. Computers & Chemistry 20, 25–33 (1996)

Opinion Based Search Ranking

Jun Jin Choong and Jer Lang Hong

School of Computing and IT, Taylor's University
{junjin.choong,jerlang.hong}@taylors.edu.my

Abstract. The Internet is one of the most widely available services in the world today. With the Internet, people are now looking for reviews on the Internet; more specifically, the social networking services. Within the social network medium, we can identify a suitable service that describes more about a person's personality as the subject. The growth of social networking popularity has contributed to the increase in information available on social networking services. The flexibility of these services allows writing individual thoughts without restrictions. With the vast information available on social networking sites today, how is it possible to look all of these opinions? How do we know which opinion holds truth? How do we know if someone is not bias based on his writing? Hence, it is seen necessary to filter opinions. In this paper we look at the possibility of using search ranking as a medium of filter opinions by exploring opinion mining methods, social networking candidates and search ranking methods. With existing sentiment analysis techniques, we can obtain opinions that are then ranked against a set of keywords.

Keywords: Search Ranking, Opinion Mining, Web Search.

1 Introduction

The idea of searching, ranking, opinion mining and social networks are not new in today's research[4]. Generally speaking the possibility of opinion mining itself is endless. With many techniques that has been thoroughly studied and prototypes built, performing search ranking on a result data mined poses a challenge that ought to be considered. For example, given a carpooling system that uses the social network, how would it be possible to know who to trust from opinions that have been written by that particular person? Is that person good or bad at handling cars? Is he or she a cautious driver? Why should I carpool with him or her? These are questions that search engines cannot answer as they are subjective in nature. This may not be the case for data mining; in particular opinion mining. For instance, in Google search ranking has proven to be a useful suggestion to answers that relate to factual questions. Most often than not, the first result from Google is usually what we are looking for. Should similar concept apply to opinion mining on social networks, we can perform conclusive searches that answer the previous questions.

If Google can answer factual questions, what about subjective questions? If we ask Wolfram Alpha a mathematical question, there would always be an answer.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 465–472, 2013.

Subjective questions may not be the same. Instead, there could be various possibilities when determining the process for answering such questions. Such complexity is defined as a complex problem in our case. A complex problem may require thorough investigation, researching before ending up on a decision. Most of the time, these are factored to scarce resources such as time and money.Assuming that most decision making can be simplified, perhaps these scarce resources could be put to better use. Therefore, we believe that the very first step for such achievement can be done with search ranking and opinion mining. Our domain of focus would be on social networks which we believe is the glue of most sparse data (opinions, reviews).

2 Related Work

2.1 Opinion Mining General Concept and Background

Opinion Mining or Sentiment Analysis is a Natural Language Processing technique (NLP) that analyses sentiment within a text [2]. Most technique revolves around the idea of sentiment polarity. This separates texts or words into positivity, negativity and sometimes neutrality. The process may look simple; but the process is complicated with many general challenges that have been investigated thoroughly by researchers. Opinion mining is subject of its own as there are many areas and possibility that needs to be covered to perform satisfactory segmentation when classifying opinions as positive or negative. Most opinion mining technique focuses on textual information, which could be classified broadly into two main categories, facts and opinions [3]. Facts are objective statements about entities and events in the world. Opinions are subjective statements that reflect people's sentiments or perceptions about the entities and events. Much of the existing research on text information processing has been (almost exclusively) focused on mining and retrieval of factual information, e.g, information retrieval, Web search, and many other text mining and natural language processing tasks. Little work has been done on the processing of opinions until only recently.

2.2 Search Ranking Using Sentiment Analysis

When information is plenty, but the search results returned are irrelevant, the search engine has failed to perform. The idea of search ranking has been created by founders of Google during their days as a PhD student; although Jon Kleinberg proposed his algorithm about the same time. Search ranking is widely used in their product, Google which is popularized by the term "Page Rank" [1], [10]. It describes the relevancy of results against the search keyword. Therefore, improving search result accuracy and plausibility; improving overall searching experience for their users. Likewise, the same concept applies on top of opinion mining. If we can identify which opinions are relevant search, we can obtain similar results. In more detail, it performs a simple filter that describes more about opinions. Taking our previous example, a carpooling system that identifies opinions of its driver and their passengers; we can identify if a driver is trustworthy or the passenger is civilized. Notice that we used terms that

describe about people. Another form of great example would be the use of search ranking to filter off irrelevant opinions. For instance, one would write reviews on a product but instead of criticizing the product, most complaints are about the publisher of the book. Such cases could still contribute to decision making, but it should not influence the outcome should someone asks "Is this book worth reading?"[7], [8].

2.3 Decision Making Support

Decision making is a complex problem in most cases. However, in our research, we would classify these problems into a much broader category; namely simple problems and complex problems. The goal of decision making support is to assist users with their decision making process. For instance, "iPhone vs Android, which is better?" would rule out to be a complex problem. To simplify things, most of our discussion focuses on a small subset of complex problems. In computer science, decision making would probably be classified under fuzzy systems or fuzzy logic. In this case however, it's not the decision of agents that we would like to assist, but rather humans themselves. If one would look up for something he/she is interested to buy and has limited resources (money), he/she would spend countless of hours before making this final crucial decision. If a bad choice is made, obviously money gone down the drain. Otherwise, the money is well spent.

3 Motivation of Using Social Network as Domain

The rise of social networking has increase the surge of interest in sentiment analysis. It is no doubt that social network has been the catalyst to sentiment analysis.Social networking services such as Facebook, Blogger, Youtube, Twitter etc. are widely used by users to share information. The flow of information propagates faster in social network services than any other comparable mediums such as printed newspapers. With speed comes at price. Information that propagates the social medium is not moderated. Hence, information written on the social network tends to be subjective in nature. Notably, this information is there to describe on a topic and share points of view with their respective audience. Due to its un-moderated nature, it is an ideal candidate for sentiment analysis.Often social networking services do not promote text searching. For instance, in Facebook searching focuses on looking up people and pages. Youtube on the other hand focuses on video title and description. There is no need to search on comments or other information. To business people, these are important factors that reflect upon their marketing strategy. "How do people like my product?" are often questions raised. Unlike reviews on niche oriented website, reviews are listed based on its usefulness to another buyer (eBay, Amazon), social networking services does not have such similarities. With exceptional cases, crowdsourcing social networking sites such as Stackoverflow and Youtube or any other sites within the Stack Exchange network has voting ranking. Unfortunately these sites are restricted to factual answers and not opinions.

4 Problem Formulation

From the concepts and definition of sentiment analysis, we get a glimpse of how sentiment analysis works. In addition, we discussed existing state-of-the-art techniques, double propagation as a method of identifying important features from a corpus. We also discussed the improvement of feature ranking mechanism used in the double propagation technique. From double propagation, we can see that the opinion lexicon is used to promote opinion mining towards a semi-supervised learning approach. Furthermore, double propagation allows us to identify the topic feature in a given corpus. The improvised feature on double propagation allows features to be more accurately extracted from the given corpus. The basis of the technique takes two different patterns namely, part-whole and "no" patterns to increase recall. It then ranks the extracted feature candidates by feature importance, which is determined by two factors: feature relevance and feature frequency. In terms of relevancy, a web page ranking algorithm was used (HITS)[5], [6]. The frequency is applied using a computational algorithm to increase the ranking of feature. The improvised double propagation has successfully increased feature extraction precision.

5 Objectives

In our problem statement, we stated that search ranking is necessary to act as a filter when using opinion mining. The idea behind this allows people to go beyond asking Google factual questions such as "Who is the prime minister of Malaysia?" instead, questions like "How does the battery lifespan of the iPhone 5 perform?" can be answered using a search ranking algorithm. Imagine, answers given back in a statistical manner. You would not need to spend time searching through thousands of newsfeed to conclude your decision for purchasing the iPhone 5. The potential of solving this problem would save each individual a lot of time for decision making. Although the domain could potentially be unlimited, to narrow down the scope, we would only discuss on social networking services.Hence, our technique needs to be as generic as possible. By the definition of generic, it should not only work on a single social networking service such as Facebook. It should ideally, be able to identify similar opinions across all social networking services and return an aggregate answer. To do so, one has to rely on method that could be plausible enough to distinguish opinion given are not fake or even bias. In addition, given that it has to work across different domains, we need to identify the common grounds between these social networking sites. We classify this as an extraction problem which will be discussed in the following section. However, in our proposed search ranking algorithm, we do not take this into account as there is only a single domain focus.

Within our objective, we will need to narrow our scope. There may be problems related to opinion mining which could affect our hypothesis, but most are out of our research scope. For instance, we need to identify a common problem within each social networking service simply because there are too many different types of social networking websites. Websites like Flickr socializes via photos and sites like Twitter

socializes via text messaging. The next problem we discovered is the language barrier. What happens if someone posts an opinion which could be useful, but contain a mixture of languages? Therefore, our technique should realize such problem exist but not to solve it. Part of this problem relates to computer vision to identify objects. This is also an extraction problem which we classify it in next section.Assuming that we have solved the smaller problems of extraction and identification of a topic feature (such as using double propagation), we need formulate a suitable ranking algorithm. One way is to use existing solutions such as HITS or PageRank[9]. Alternatively, we could propose an improvement on existing solutions but adding additional vectors into consideration in the ranking algorithm. In the following section, we take a look at a few proposed examples that we think are important in addition to frequency and relevancy.

6 Proposed Solutions

6.1 Overview

With HITS as the basis for our search ranking algorithm, we include an additional vector that can be obtained from opinion mining. This vector includes the overall polarity (positivity, negativity or neutrality) of a given corpus. To obtain this vector, an initial run of sentiment analysis algorithm is done on the given corpus. The result returned is the polarity of the corpus.In relations to HITS which uses authority and hub to rank the importance of a given page, we need to translate such vector from opinion mining to HITS [13]. In our solution we would classify an opinionated text as hubs and the authority as the owner of the text. There is more than one way to do this. For most social networking sites, a single user may have different types of friends. Among these friends, some would be reputable, smart, or possibly unknown. If taken into consideration of such status, we can formulate a similarity between HITS algorithm and our opinion based search ranking algorithm. Alternatively, we can also consider the number of shares, likes etc. Although useful, there is a sign of bias vector being included in the algorithm. What happens if an opinionated text is written with honesty from an unknown individual compared to popular individuals who wrote short and biased opinions? Truthfully, the popular individual will be given more weightage in terms of authority. In most cases, PageRank works the same way but with additional vectors such as content quality being a consideration. PageRank would give higher ranking to well establish sites like bbc.co.uk and then consider the links in that site (hubs) as important.

Unlike Google's PageRank, we cannot take quality of content as a vector; rather we consider the overall sentiment polarity as sufficient vector to denote that an opinion is of higher quality. Since polarity can be higher or lower, our next step is to normalise it. Consider that we have a total polarity equating to 1. We would consider 0.5 being neutral, < 0.5 being negative and > 0.5 being positive. The calculation to perform the normalisation would be dependent on the type of opinion mining method used and its accuracy.Next, we would need to consider the scaling factor. Should the scale of opinion increase, we need another way to determine the existing standings of

previous opinions. In this case, we would follow HITS algorithm. For each iteration, we will compute a normalisation value for the authority and the hub. This would create a converging value. The iteration would stop upon reaching a converging value that is negligible of changes.

6.2 HITS Algorithm Revisited

To understand how our algorithm works, we need to reiterate the importance of HITS algorithm. Using HITS idea of authority and hubs, we can relate to which opinion carries higher importance over another opinion. The idea behind authority and hubs is a mutually reinforcing relationship. In the case of opinion mining, the terms are changed. A good hub is a comment/feed points to many good authorities; a good authority is a comment/feed that is pointed to by many good hubs.For example, consider the case where a comment on iPhone battery lifespan. John is a highly reputable user on the social network having many friends or followers comments on the iPhone battery lifespan. His comments are "retweeted" by many of his followers (Twitter) or shared by many friends (Facebook). The mutually reinforcing relationship states that such condition would provide John's opinion with higher weightage. Obviously, the vector to determine whether what is an authority and hub can differ in different social networking sites. In this case, Facebook could be number of shares, comments, likes or in the case of Twitter; it could be the number of retweets, or replies.

The method we used to determine the authority or hub is by using HITS algorithm. More precisely, we take the sum of authority and hub score from each node and compared them to see which is higher. The higher score for authority vector would be the authority node.By using HITS algorithm as our foundation, we have successfully connected the search term a relevancy. That is, given that a user has so many news feed or comments, we are likely to know which opinion we are looking for. In other words, we have performed a filter on all opinions that could be related to the search query.HITS algorithm has its limitations if we consider its usage in opinion based search ranking. We need to provide the algorithm with an extra vector that justifies this weakness. Hence, in the section, we propose simple strategy by first deducing the overall polarity of the opinionated text.

6.3 Sentiment Polarity as Vector

One of the benefits when using opinion mining is the underlying technique used. Opinion mining simplest technique could classify a given text in its overall sentiment polarity form given an opinionated text. In questions such as "Is the iPhone battery durable and lasting?" which we have used throughout our example, HITS alone was insufficient to justify a decision making. Looking back at the previous section, we conclude that an authority node is a highly trusted opinion. Subsequent shares, likes, comments from it would reinforce the relationship between the authority and hubs. What is missing is the weightage of a given opinion. With weightage taken into consideration, we can make decisions on opinions that are old, fake or even biased. Overall, we are only interested in the output value from the opinion. Additional factors

would be included as considerations in an opinion mining technique; a filter in opinions itself.For instance, given the example text "iPhone has tremendous battery life!" we would run a sentiment analysis on it. Assume the function *f(opinion, e)* where *e* is the optional parameter for consideration such as date, length of opinion etc. The function should return a weightage in the form of normalised vector; between 0 and 1.

Equation 1: $f(opinion, date, length) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Where *x, y* and *z* denotes the normalised value for opinion, date and length respectively such that *x* would denote the overall polarity of the text ($x > 0.5$ for positivity, $x < 0.5$ for negativity or *0.5* for neutrality). The remaining variables *y* and *z* could be a scale determine by the search engine. For example, saying $y = 0.8$ would mean that the date is of *80%* accurate. Alternatively we could say that the importance of the date hereby can be determined by a scale of *0.8*. The vector produced by the function would return 0.9 which denotes the positivity of the opinion, y which determines if the date is within exactly 2013-05-01 and finally z which tells the function that length can be anything with the input value being 0.This function provides a simple example on how we can go about identifying the weightage of the opinion with other factors into consideration. The assumption of implementing this function is such that there is a way to determine the polarity of the opinionated text. The method of determining is taken into consideration during opinion mining. Therefore, the additional overhead may be negligible.

7 Conclusions

The aim of this paper is to hypothesise a possibility such that opinions can be ranked given the usage of opinion mining techniques either by indexing or real-time. The algorithm may or may not proof to be useful, it shows that there is a possibility of requiring ranking algorithms with the boom of user generated content on social networking mediums. Ideally, we would like to be able to ask the search engine and conclude our decision immediately (decision making support). By taking the very first step of linking both opinion mining and search ranking, we can see that there is a possibility to produce a system that could truly be of use to save time and money.In future possibilities, we look forward to investigating alternative search ranking algorithms that could potentially show its usefulness with opinion mining. Perhaps an algorithm that goes beyond considering authority and hubs or even probability of a user clicking a particular link.

References

1. Altman, A., Tennenholtz, M.: Ranking systems: the PageRank axioms. In: Proceedings of the 6th ACM Conference on Electronic Commerce, pp. 1–8. ACM, New York (2005)
2. Liu, B.: Sentiment analysis and subjectivity. In: Handbook of Natural Language Processing, 2nd edn. Taylor and Francis Group, Boca (2010)

3. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL 2002 Conference on Empirical Methods in Natural Language Processing, vol. 10, pp. 79–86. Association for Computational Linguistics, Stroudsburg (2002)
4. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: Computer Networks and ISDN Systems, pp. 107–117. Elsevier Science Publishers B. V. (1998)
5. Farahat, A., LoFaro, T., Miller, J.C., Rae, G., Ward, L.A.: Authority Rankings from HITS, PageRank, and SALSA: Existence, Uniqueness, and Effect of Initialization. SIAM Journal on Scientific Computing 27, 1181–1201 (2006)
6. Kleinberg, J.M.: Hubs, authorities, and communities. ACM Comput. Surv. 31 (1999)
7. Langville, A.N., Meyer, C.D.: Deeper inside pagerank. Internet Mathematics 1, 2004 (2004)
8. Langville, A.N., Meyer, C.D.: Google's PageRank and beyond: the science of search engine rankings. Princeton University Press, Princeton (2006)
9. Li, L., Shang, Y., Zhang, W.: Improvement of HITS-based algorithms on web documents. In: Proceedings of the 11th International Conference on World Wide Web, pp. 527–535. ACM, New York (2002)
10. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bring-ing Order to the Web, `http://ilpubs.stanford.edu:8090/422/`

A Service-Oriented Approach with Neural Networks and Knowledge Ontologies for Robot Control

Tsung-Hsien Yang and Wei-Po Lee

Department of Information Management,
National Sun Yat-sen University, Taiwan
{D954020005,wplee}@mail.nsysu.edu.tw

Abstract. Researchers have been building robots able to interact and work with people at home. To share and reuse robot code between different developers, we present a service-based approach that exploits the standard web interface to create reusable robotic services. Our approach includes knowledge ontology planning and neural network learning strategies for robot control. In addition, several service functions, including service discovery, selection, composition, and reconfiguration have been developed for operating the services. The proposed approach has been implemented and evaluated. The results show that our approach can be used to build robotic services successfully.

Keywords: service-oriented computing, robot control, neural controller, knowledge ontology, AI planning, service composition.

1 Introduction

Building home service robots is a significant issue and researchers have now been developing robot design frameworks to manage the complexity and facilitate the reusability of robot code (e.g., [1-2]). From the point of view of the end-users, they expect to obtain and use robot application software conveniently; on the other hand, from the point of view of the robot designers, they prefer an easy-to-share environment in order to integrate application services constructed by different providers. Taking into account the needs of both sides, service-oriented architecture (SOA, [3]) provides a promising option for developing robotic services. Ideally, with a SOA-based robotic framework, end-users can control their robot in the similar way of using web services, and robot designers can share services with each other and reuse available code to design more comprehensive services.

However, unlike the traditional web services, applying SOA to robot applications involves the complicated control of robot actions, and thus the design of robotic services becomes challenging. To develop robot code to solve complicated application tasks, a common way is to adopt a divide-and-conquer strategy to reduce the task complexity. The process of dividing and solving robotic task is similar to that of task decomposition and service composition in the web service domain, in which most problems can be solved by workflow-based or AI planning methods [4-5]. Hierarchical Task Network (HTN) planners are typical examples [6]. With the aid of an

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 473–483, 2013.

ontology defined to describe robot tasks and to guide the task decomposition, the methods used in web composition can also be applied to developing of composite robotic services to solve complicated tasks.

In the study of service-oriented computing and modeling, web service technologies have been widely applied to different types of applications. With many successful experiences, researchers are now pressing on to extend these techniques to the development of robot systems. The most related works are the ones regarding robots as services and exploiting the web service architecture to create robots. For example, Yachir *et al*. employed service composition techniques to plan robotic services to help an elderly person [7]. Kim *et al*. focused on how to control a robot through the integration of web service and robot application technologies [8]. Similarly, Ha *et al*. proposed a service-oriented architecture for the integration of ubiquitous robot devices, including sensors and motors [9]. In addition, Osentoski and colleagues have built a framework to enable web interfaces for robot code sharing [10].

Different from the above studies, our work dedicates to construct the explicit task ontology for daily home tasks, and focuses on the planning-based service configuration to obtain robotic services. In particular, we develop a practical method to create new services and this issue has not been addressed in the work described above. Our method is a kind of demonstration-based programming [11], which is to teach a robot how to achieve a task through human demonstration. This method represents a task as the behavior trajectories at the data level, and it includes a neural network learning procedure to derive controllers. Once robot controllers are created and regarded as services, our framework integrates these services with a HTN planner to perform ontology-based service composition to achieve more complicated tasks. In addition, we develop an adaptive mechanism with a service re-planning procedure to modify the incorrect or unexpected robot action sequences during the composition process. To verify the proposed system, we have conducted different sets of experiments in which the robot can successfully achieve user-specified control tasks in a home environment.

2 A Service-Oriented Approach for Robot Control

Service-oriented computing (SOC) has been widely used to support the development of software applications. Following the SOC design principles, we present a service-oriented robotic framework (as illustrated in Fig. 1) to provide rapidly prototyping robotic services. As can be seen, our work mainly includes knowledge ontologies and three service modules. The task ontology shown in the figure is built for command interpretation and service mapping. It describes the semantics of a task in terms of task structure and task-solving process. The service modules are a machine learning mechanism for the creation of new services, a service discovery module (that analyzes the user command, searches the service repository, and chooses the most suitable service), and a service composition module (that finds relevant services and integrates them to achieve the target task). The details are described in the subsections below.

2.1 Using Knowledge Ontology to Identify Robot Task

Ontology plays an important role in knowledge representation. For an action planning problem, ontology can be defined to provide the sequence of problem-solving steps through organizing domain knowledge, and then the planner can exploit domain knowledge and the related techniques defined within the ontology to achieve the application task. In this work, we construct two ontologies, task ontology and position ontology, to specify the structure of the problem-solving process and to describe the environmental knowledge for the robot, respectively. To accomplish the target task, a service robot should understand both ontologies.

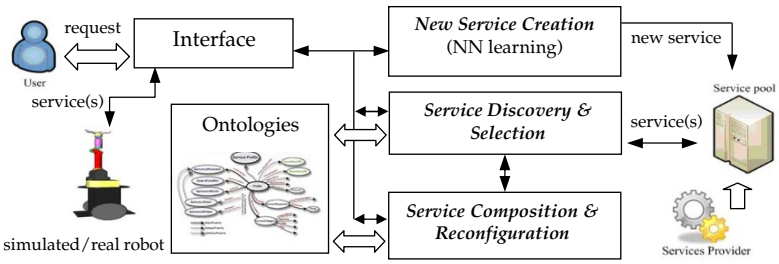


Fig. 1. Schematic diagram of the proposed system

The first ontology, position ontology, defines the locations of different objects in a home environment and the containment relationships between the objects. For example, this ontology indicates that the bowls, plates, and cups are put on a cupboard, and a TV is placed in the living room. And the second ontology, task ontology, describes how to resolve a user's request by a sequence of steps. It interprets the task complexity in a hierarchical way. The task ontology has included the possible robot actions/tasks in the environment so that the robot can follow it to achieve the target task.

The procedure for using task ontology is that, after receiving a user command, the system will parse the command to extract the verb part as the action description, and then use the description to match the task terms recorded in the task ontology. Currently, we take a simplified natural language toolkit (i.e., NLTK, [12]) for command parsing. For example, a command "give me a cup" will be analyzed to get the verb "give", and then use the word "give" to match the terms included in the task ontology.

In our framework, we use the powerful ontology description language OWL-S to implement the ontology. It has three major parts with essential types of knowledge: *service profile*, *service model*, and *service grounding*. In general, service profile provides the information to express "what the service does"; service model describes "how to use the service"; and service grounding presents "how to access a service". Fig. 2 shows an OWL-S example that describes the service model defined for the robot behavior "Give", which is composed by one atomic process "Find" and two composite processes "Get" and "Put". The details are described in section 3.2.

```
<process:CompositeProcess rdf:ID="Give">
  <process:hasInput rdf:resource="#Person"/>
  <process:hasInput rdf:resource="#Object"/>
  <process:hasPreconditoion rdf:resource="#ExistPerson"/>
  <process:hasPreconditoion rdf:resource="#ExistObject"/>
  <process:hasEffect rdf:resource="#PersonHasObject"/>
  <process:composedOF>
    <process:Sequence>
      <process:components rdf:parseType="Collection">
        <process:CompositeProcess rdf:about="#Get"/>
        <process:AtomicProcess rdf:about="#Find"/>
        <process:CompositeProcess rdf:about="#Put"/>
      </process:components>
    </process:Sequence>
  </process:composedOf>
</process:CompositeProcess>
```

Fig. 2. The OWL-S example of a robotic service model

2.2 Learning Neural Controllers

The core of creating a new service is to develop the service function, which can then be combined with other relevant service descriptions and specifications to constitute a service. On the robotic service domain, the service function means the control code for driving the robot to achieve a task. The code can be written manually by a programmer or alternatively be learnt without explicit programming.

To create new services, here we adopt the demonstration-based approach we developed previously for robot learning ([13]), and then pack the control code to work as the service functions. In this approach, perception and motion information of the behavior sequences demonstrated by the user are first recorded, and then a machine learning mechanism is used to derive controllers. Fig. 3 illustrates the procedure. At first, the robot is driven manually to achieve the target task. In this stage, the robot is regarded as a teacher showing the correct behavior. During the demonstration period, the relevant information is recorded to form a data set for later training. In other words, it is to derive the time-series profiles of perception and motion information from the qualitative behavior demonstrated by the robot. After that, in the second stage, the robot plays the role of a learner that is trained to achieve the target task.

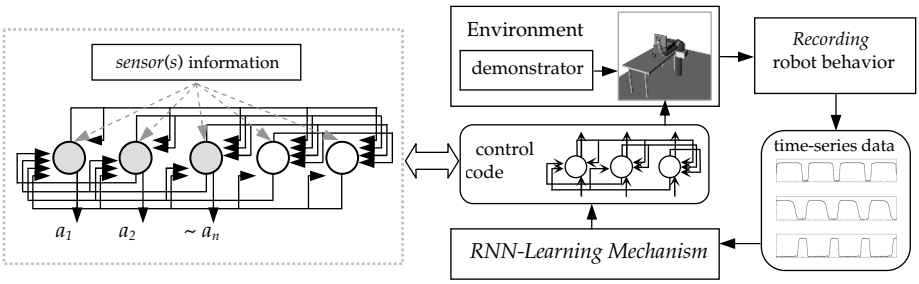


Fig. 3. The learning procedure and the neural network controller for the robot

In this work, a recurrent neural network (RNN) model is adopted as a behavior controller for the learner. Here, we take a fully connected RNN architecture as the robot controller, and implement a learning mechanism to train the controller. In a fully recurrent net, each node has a link to any node of the net, including itself. When activity flows through the network in response to an input, each node influences the states of all nodes in the same net. The regulatory effect toward a node is transformed by a sigmoidal transfer function into a value between 0 and 1 for normalization.

When the above model is used to control a robot, each network node corresponds to an actuator of the robot in principle. Also, two extra nodes are added to serve as buffers, and their roles are not specified in advance. The redundancy makes the controllers easier to be learnt from data. Fig. 3 (left) illustrates the architecture of our robot controller. In this architecture, the sensor information received from the environment is continuously sent to all nodes of the fully interconnected network, and the outputs of the actuator nodes are interpreted as motor commands to control the robot. To find the settings of a neural network, we adopt the back-propagation through time learning algorithm (BPTT, [14]) to update the relevant parameters. The goal here is to minimize the accumulated discrepancy between the time-series data recorded in the demonstration procedure (i.e., desired values) and the values produced by the control model (i.e., actual values). Hence, the above error function is defined as the mean squared error over the time period:

$$\sum_{i=1}^{N}\sum_{t=1}^{T}\left\{\frac{x_i^a(t)-x_i^d(t)}{x_i^d(t)}\right\}^2 \tag{1}$$

In the above equation, $x_i^d(t)$ is the desired output of node i at time t, $x_i^a(t)$ is the value generated from node i of the learnt model, N is the number of nodes in the network, and T is the number of time points for data collection.

2.3 Service Functions

In this work, four types of functions, including service discovery, selection, composition, and reconfiguration, are developed to launch the framework. The first type of functions is service discovery. As mentioned, we use OWL-S to describe services, so an automatic method can be developed to find appropriate services. In our system, the OWL-S service profile defines the IOPE elements (i.e., Input, Output, Precondition, and Effect), which are used to match the user's service requests. A successful match means that the system can find services (i.e., find the task terms defined in the ontology) to fulfill the user's request. More comprehensive method (e.g., Jaccard coefficient) can be taken and modified slightly as similarity calculation, based on the name, essential information, and relationship defined in the ontology.

The second type of functions, service selection, is to choose the most suitable services from the candidates obtained from the service discovery process. To perform service selection, many researchers have defined various attributes (such as cost,

response time, reputation, etc. [5]) and proposed different quality-of service (QoS) based methods. Among these attributes, service reputation represents the direct evaluation results from the service requesters or the neutral party. It is an objective and easy-to-measure attribute. Therefore, we choose to use this attribute for service selection, and design a rating strategy to measure service reputation. It is to combine all ratings given by users to rank the candidate services. The system will then select services according to the ranking order, in response to the user's request.

The third type of functions is service composition, which is to automatically compose services already existing in the repository to complete more complex tasks. In our framework, we adopt the AI planning techniques to achieve service composition. Among others, HTN planning is a well-designed methodology most suitable for the service composition, in which the tasks are categorized into two types: the primitive and the compound. A primitive task can be performed directly by the predefined planning *operators*, while a compound task needs to be decomposed by a planning *method* before being performed. The concept of task decomposition in HTN is in fact very similar to the composite process decomposition in the OWL-S service model: it reduces the complexity of reasoning by eliminating uncertainty during the planning process. One well-implemented HTN planner is the Simple Hierarchical Ordered Planner 2 (SHOP2, [6]). Following their studies, we choose to employ SHOP2 to conduct our service composition and take the task ontology defined above to guide the decomposition. It should be noted that as OWL-S and SHOP2 have their own internal representations, to use SHOP2 for service composition, translations between OWL-S and SHOP2 need to be defined. Sirin *et al*. have defined some translations between them [6]. Thus, a service originally created by the OWL-S descriptions can have its SHOP2 format and be used by the planner for service composition.

The final type of functions is service reconfiguration. Though the above service functions can offer the user a set of services for his task, however, due to some unforeseen environmental situations, in some cases the robot cannot achieve the task by the services organized by the planner. To remedy this problem, our system performs a re-planning procedure iteratively, by accounting for the newly acquired world states. The system will highlight the planning steps related to the incorrect behavior sequences, so that the user can inspect the highlighted parts to find out the failure, modify the world states, and then activate the planer again to find other services to substitute the current ones. The user can also create new services to fix an incomplete plan through the demonstration-based mechanism described previously.

3 Experiments and Results

3.1 Service Creation through Robot Learning

To evaluate our framework, two sets of experiments have been conducted to investigate whether the neural controllers can be built through the learning module, and to

examine how our service modules can be used to achieve a more complicated task at a higher level. In the first set of experiments, a robot arm with a camera has been used to learn a controller for a sequential task of opening a box and picking-up a cup inside the box. To save evaluation time, the experiments were performed in simulation.

The learning module has been used to achieve a sequential task that involves the integration of perception information and internal state of the controller for action selection. Initially, a box was put on the table. The robot needed to open the box and decided what to do next according to what it observed. If the robot found a cup in the box, it was required to pick up the cup, put the cup on the table, and then close the box. If the box was empty, the robot simply closed the box. After the box was closed, the robot had to decide to take which of the following two actions: whether to open the box (in order to check the box and pick up the cup if there was any) or to move back to its initial position (the task had been completed). To make the correct decision, the robot needs to integrate both the perception information and the previous task.

The control mechanism for the above sequential task can in fact be considered as a finite state machine that includes four internal states to represent respectively the task status below: (1) the box is on the table and the robot is in its initial position; (2) the box is on the table and robot has moved to the position ready for operation; (3) the box has been opened by the robot; and (4) the box has been closed by the robot. For this task, the perception input to the finite state machine is one of the three situations (the box is close, the box is open and the cup is in the box, and the box is open and it is empty) derived from the visual results that have been processed by the camera on the robot arm. Therefore, the control task is to deal with the state transitions based on both the current state and sensor information, and then to generate an appropriate action. Here, the possible actions for the robot are to move to the position ready for operation, to open the box, to pick up the box, to close the box, and to move to the initial position. To obtain a controller that can produce the expected sequential behavior, the proposed approach has been used to learn a six nodes network. Fig. 4 illustrates the behavior generated by the controller in the robot software OpenRAVE [15]. As we can see, a sequential controller can be learnt to achieve the task successfully.

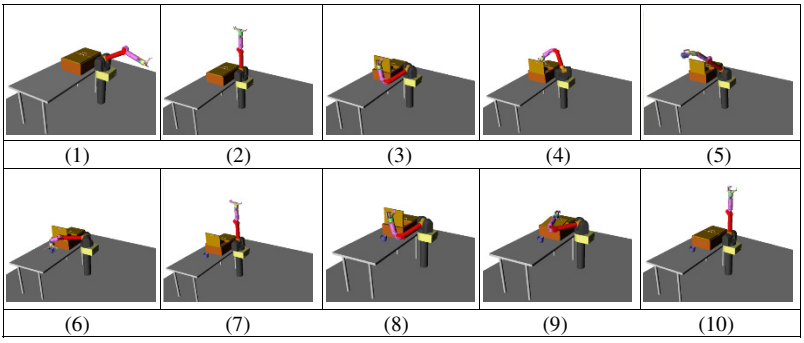


Fig. 4. The behavior sequence produced by the neural controller learnt

3.2 Service Composition through Task Planning

In the second series of experiments, we use our framework to derive a composite service to achieve an application task: the user asked the robot to give him a cup in a home environment. Two test scenarios have been arranged for this task. In the first case, the cup is placed on the table and it is visible to the robot; while in the second case, the cup is moved to an opaque cupboard so that the robot must find the cup before picking it up. The former is to verify whether the system can find a suitable service to drive the robot to pick up the cup, and the latter, to examine whether the system can make a new plan to fulfill the target task in a changed environment, by retrieving available services (controllers) or creating composite ones.

To understand the user's request, the robot needs to connect to the ontology server for parsing the command "give me a cup". In this application, the system used a natural-language parser to check the syntax of the command and generate a syntax tree. It then sent the verb part to the task ontology and the noun part to the position ontology to search for suitable services. The OWL-S example shown in Fig. 2 describes the service model defined for the robot behavior "Give", which is composed by one atomic process "Find" and two composite processes "Get" and "Put". That is, the task "Give an object to a person" can be decomposed into three subtasks: "Get the object", "Find the person", and "Put the object to the person (meaning location here)". In addition, the process for "Give" has two inputs "Person" and "Object" (to tell the robot which object to take, and to whom) and two preconditions "ExistPerson" and "ExistObject" (to indicate if any person or object exists in the world state).

With the decomposition result, the system can translate the "Give" service described by OWL-S into a HTN method, and invoke the sub-services included in the "Give" process to complete the overall task. Fig. 5 shows the HTN method corresponding to the "Give" service, which includes the subtasks "Get", "Find", and "Put". The action steps of "Give me a cup" can be carried out by the relevant atomic services (i.e., HTN operators), in the order of the following verb-object pairs: (find cup), (move table), (grasp cup), (find user), (move user), and (release cup).

Fig. 6 illustrates how the robot achieved the task in the two test scenarios in simulation. The first three steps in the figure (i.e., steps (1)-(3)) show that the robot used the position ontology to infer where the cup was located. For the first scenario, once the robot knew that the cup was in the kitchen, it went there and recognized that the cup was put on the table. Then the robot moved to the region around the cup so that it could grasp the cup. Steps (4)-(6) describe such a situation: the robot moved to a position close to the cup, and performed the "Get" service. Next, steps (7)-(10) show that after the robot picked up the cup, it invoked the other two services "Find" and "Put" to find the user and give him the cup.

As mentioned, in the second test scenario, the cup was moved to an opaque cupboard, and the robot could not grasp the cup directly (though it could obtain the cup's position from the environment file and move to a position around the cup).

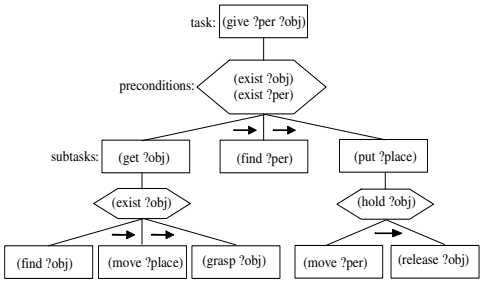


Fig. 5. The HTN method for the target service

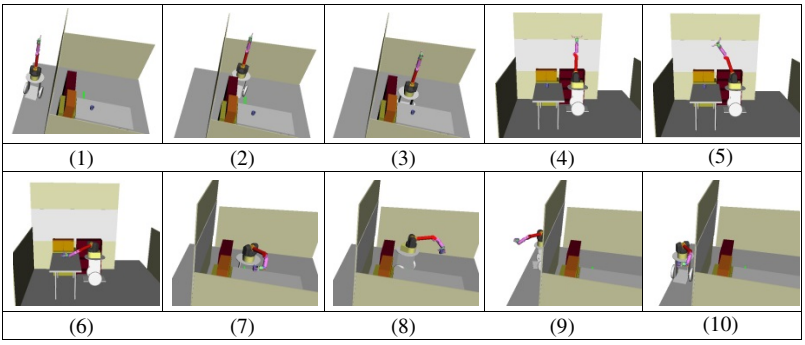


Fig. 6. Simulation results of the first case

This means that the current service could not fulfill the user's requirement, as the service "find" in Fig. 7 could not result in any state "found object" which was the precondition of the service followed. To cope with such failure, the system highlighted the relevant steps in the HTN plan, so that the user could inspect this part to find out the reasons that caused the failure, modify the world state, and then make a new plan.

In the experiment, the user found that the cup did not appear in a visible place so the robot could not complete the subtask "grasp". To solve this problem, the user changed the world state from "cup is visible" to "cup is invisible", and then used the HTN planner again to make a new plan. That is, the user added a new world state "(isContainer cupboard)" as the precondition and a new input "cupboard" to the planner to find suitable services. After that, the system found a "search" service (to open containers to check if any of them contains the target object) that matched the new requirement. As shown in Fig. 7, the new plan also indicated that this "search" service must be performed before the "grasp" service.

Taking the new plan, the robot first checked the position ontology to obtain the possible cup position, and it then examined the world states to find the objects "containers" which could be used to store the cup. For any object with a property "container", the robot activated the service "open" to open it, and checked if the cup was inside there. In this way, the robot realized that the cup was placed in the cupboard

and it confirmed that the world state "(isContainer cupboard)" was true. According to the other world state "(isClosed cupboard)", the robot tried to open the cupboard, and then grasped the cup successfully. Fig. 8 shows to the above process: the robot opened the cupboard and grasped the cup to achieve the task successfully.

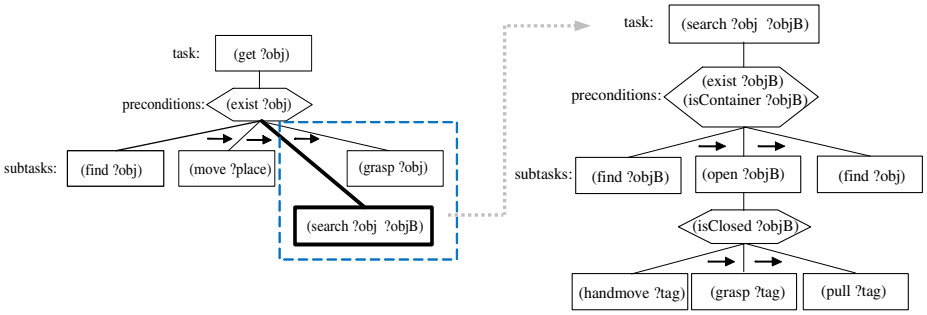


Fig. 7. Reconfigured plan for the target task

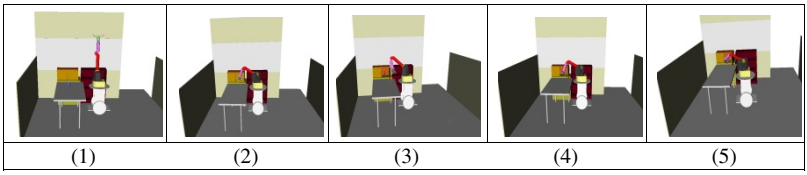


Fig. 8. Simulation results of the second case

4 Conclusions

In this work, we indicated the importance of developing an easy-to-share networking platform for the reuse of robot code distributed by different providers. Taking into account the needs of both sides of end-users and robot designers, we presented a service-oriented approach that exploits the standard web interface for the development of shared robotic services. Our approach includes knowledge ontology and neural network learning strategies for robot control. The task ontology has been constructed and used for command interpretation and service mapping, and the neural network has been used to create new services through a learning procedure. In addition, several service functions have been developed for operating the services. Under the guidance of task ontology, our work employs a planning-based service composition process to generate composite robotic services to solve complicated tasks in the home environment. Experiments have been conducted to verify the proposed methodology, and the results show that new services can be built through human-machine interaction and composite services can be derived for the application task successfully. Currently, we are extending this robotic service approach to investigate more comprehensive service mapping and composition strategies to take more task-related issues into account to infer composite services precisely.

References

1. Schiffer, S., Ferrein, A., Lakemeyer, G.: CAESAR: An Intelligent Domestic Service Robot. Intelligent Service Robotics 5, 259–273 (2012)
2. Beetz, M., Jain, D., Mösenlechner, L., Tenorth, M., Kunze, L., Blodow, N., Pangercic, D.: Cognition-Enabled Autonomous Robot Control for the Realization of Home Chore Task Intelligence. Proceedings of the IEEE 100, 2454–2471 (2012)
3. Erl, T.: Service-Oriented Architecture (SOA): Concepts, Technology, and Design. Prentice Hall (2005)
4. Kuzu, M., Cicekli, N.K.: Dynamic Planning Approach to Automated Web Service Composition. Applied Intelligence 36, 1–28 (2012)
5. Luo, S., Xu, B., Yan, Y., Sun, K.: Towards Efficiency of QoS-Driven Semantic Web Service Composition for Large-Scale Service-Oriented Systems. Service Oriented Computing and Applications 6, 1–13 (2012)
6. Sirin, E., Parsia, B., Wu, D., Hendler, J., Nau, D.: HTN Planning for Web Service Composition Using SHOP2. Web Semantics: Science, Services and Agents on the World Wide Web 1, 377–396 (2004)
7. Yachir, A., Tari, K., Chibani, A., Amirat, Y.: Towards an Automatic Approach for Ubiquitous Robotic Services Composition. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3717–3724 (2008)
8. Kim, B.K., Miyazaki, M., Ohba, K., Hirai, S., Tanie, K: Web Services Based Robot Control Platform for Ubiquitous Functions. In: Proceedings of IEEE International Conference on Robotic and Automation, pp. 691–696 (2005)
9. Ha, Y., Sohn, J., Cho, Y., Yoon, H.: A Robotic Service Framework Supporting Automated Integration of Ubiquitous Sensors and Devices. Information Sciences 177, 657–679 (2007)
10. Osentoski, S., Jay, G., Crick, C., Pitzer, B., DuHadway, C., Jenkins, O.C.: Robots as Web Services: Reproducible Experimentation and Application Development Using rosjs. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 6078–6083 (2011)
11. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A Survey of Robot Learning from Demonstration. Robotics and Autonomous Systems 57, 469–483 (2009)
12. Loper, E., Bird, S.: NLTK: the Natural Language Toolkit. In: Proceedings of the ACL 2002 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, pp. 63–70 (2002)
13. Lee, W.P., Yang, T.H.: Combining GRN Modeling and Demonstration-Based Programming for Robot Control. Neural Computing and Applications 20, 909–921 (2011)
14. Werbos, P.J.: Backpropagation through Time: What it does and How to Do it. Proceedings of the IEEE 78, 1550–1560 (1990)
15. Diankov, R., Kuffner, J.: OpenRAVE: a planning architecture for autonomous robotics. Technical Report CMU-RI-TR-08-34, Robotics Institute, Carnegie Mellon University (2008)

A New Gene Expression Profiles Classifying Approach Based on Neighborhood Rough Set and Probabilistic Neural Networks Ensemble*

Jiang Yun, Xie Guocheng, Chen Na, and Chen Shan

College of Computer Science and Engineering, Northwest Normal University, 730070, Lanzhou Gansu, P.R. China
{jiangyun,chengguoxie,chenna,chenshan}@nwnu.edu.cn

Abstract. With the advancement of bioinformatics, the research on the gene chips has been paid more attention by the researchers in recent years. Applications of gene expression profiles on cancer diagnosis and classification have gradually become one of the hot topics in the field of bioinformatics. According to the gene expression profiles characteristics of high dimension and small sample set, we propose a classify method for cancer classification, which is based on neighborhood rough set theory and probabilistic neural network ensemble classification algorithm. Firstly, genes are sorted by using Relief algorithm. Then, classification informative genes are selected using the neighborhood rough set theory. At last, we do cancer classification with probabilistic neural networks ensemble classification model. The experimental results show that the proposed method can effectively select cancer genes, and can obtain better classification results.

Keywords: Gene Expression Profiles, Neighborhood Rough Set, Probabilistic Neural Networks Ensemble Classification Algorithm.

1 Introduction

DNA microarray is a new technology which is an efficient detection of DNA sequence. The researchers used microarray technology have obtained the expression levels of tens of thousands of genes, resulting in large scale gene expression profiles. It is very important significance by gene expression profiling for cancer diagnosis and treatment. In addition, it is also one of the important subjects of current research in bioinformatics[1]. In fact, only a small number of genes are really related to the same sample categories. These genes are referred to as the information gene. They contain

* This work was partially supported by the National Natural Science Foundation of China (NSFC) under grant No. 61163036, No. 61163039. The Natural Science Foundation of Gansu under grant No.1010RJZA022 and No.1107RJZA112. The Fundamental Research Funds for Universities of Gansu Province in 2012. The Foundation of Gansu University Graduate Tutor No.1201-16. The third Knowledge Innovation Project of Northwest Normal University No.nwnu-kjcxgc-03-67.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 484–489, 2013.

the classification information of the samples[2]. How to use the microarray data to identify feature genes, this study has proposed the new task of data mining, and became a hotspot in the research of gene expression data processing and analysis[3-4].Domestic researchers have conducted more comprehensive and detailed review of the results of these studies[3-5]. In order to select feature genes, we propose a feature gene selection algorithm based on neighborhood rough set. By using this method, we can select a relatively small number of feature genes with strong classification ability. Recently, researchers begin to pay more attention to the ensemble classification method which has higher classification accuracy and better generalization ability[6]. Hansen and Salamon proof that simply by training multiple neural networks and integration of their results can significantly improve the generalization ability of the neural network[7].Tan et al. concluded that the ensemble method can improve the classification accuracy of gene expression data [8]. In [9], A.Ben-Dor examined the use of scoring methods, measuring separation of tissue type using individual gene expression levels. In [10], a support vector machine and the decision tree were employed. For cancer classification problem, combined with the idea of ensemble learning, we propose bagging-based probabilistic neural network ensemble classification algorithm for classification of cancer gene expression profiles. To compare with the [9] and [10], our experimental results show that the proposed method can effectively reduce the classification errors, improve the classification accuracy and has better generalization ability.

The following sections of the paper are organized as follows. In section 2, we outline the methodology of feature gene selection. In section 3, we present probabilistic neural networks ensemble classification method. In section 4, we present gene expression profiles classification method based on neighborhood rough set and probabilistic neural networks ensemble. In section 5, we present experimental results and analysis. Finally, in section 6, we show our conclusions and future work.

2 Feature Gene Selection

The high dimensionality of the gene and the high noise, redundancy which the gene contains is the main reason for the low efficiency and poor performance of a variety of classifiers in the analysis of cancer gene expression data. Relief algorithm[11] as a kind of sorting algorithm has been widely used in feature selection. But, there is still a high degree of redundancy and noise among genes selected by using it.

The rough set theory is proposed by Pawlak to deal with ambiguity and uncertainty of knowledge[12]. The gene expression value is a continuous real data while Pawlak rough set cannot directly deal with. Neighborhood rough set is developed on the basis of the classical rough set theory model to deal with continuous data directly. Cancer classification problem can be formalized expressed as a neighborhood decision table $NDT =< S, A = G \bigcup D, V, f >$. $S = \{s_1, s_2, \cdots, s_m\}$ is a non-empty set of cancer samples. $G = \{g_1, g_2, \cdots, g_m\}$ is a non-empty gene subset. $D = \{L\}$ is an output feature vector. L represents sample category tag. V_a is the range of the property $a \in G \cup D$.

And f is an information function which can be expressed as $f: S \times (G \cup D) \to V$. If $\forall s_i \in S$ and $B \subseteq G$, and the neighborhood of samples s_i in the sub gene space B are denoted by $\delta_B(s_i)$, there is $\delta_B(s_i) = \{s_j \mid s_j \in S, \Delta_B(s_i, s_j) \le \delta\}$ accordingly.Typically, there are three widely used measure functions: the Manhattan distance, Euclidean distance and the Chebychev distance. Set s_i and s_j are two samples of n dimensional gene space $G=\{g_1, g_2, \cdots, g_n\}$, $f(s, g_i)$ is the value of the sample s in the i th dimension gene. Then the Minkowsky distance can be defined as following (1):

$$\Delta_p(s_1, s_2) = \left(\sum_{i=1}^{n} |f(s_1, g_i) - f(s_2, g_i)|^p \right)^{1/p} \tag{1}$$

Given a neighborhood decision table NDT, $\{X_1, \cdots, X_c\}$ is a sample subset which has specific category value of the decision attribute from 1 to c. Where $X_i \cap X_j = \varnothing, i, j \in [1, c], i \neq j, \bigcup_{i=1}^{c} X_i = S$, $\{X_1, \cdots, X_c\}$ is a division of S. $\delta_B(x_i)$ which is produced by the gene subset $B \subset G$ is neighborhood information granularity including a sample x_i.

The gene selection method based on attribute reduction algorithm is to obtain a feature gene subset as small as possible while its classification ability will be as strong as possible. Here, we use the attribute reduction algorithm based on neighborhood rough set model which is FARNeM[12]. But, because of the influence from cancer unrelated genes and noise, the classification performance of the genes reduction subset obtained in this way is not very good. Accordingly, using the Relief algorithm to sort genes and select initial feature subset is very important.

3 Bagging-Based Probabilistic Neural Networks Ensemble Classification Method (Bagging-PNN)

In practical applications, Probabilistic Neural Networks(PNN)[13] has the advantage of using linear learning algorithm to complete the work which is ever done by nonlinear learning algorithm. Such networks corresponding weights are related to the distribution of model samples. The network does not require repeated training. It selects data separately from the original data set randomly. By giving a weak learning algorithm, bagging technology can learn multiple training sample set. The results with the largest number of votes will be the final results. Bagging-PNN Ensemble Classifier Algorithm is as following.

Algorithm: Bagging-PNN Ensemble Classifier
Input: *D*--training dataset; *T*--number of individual classifiers;
A--Probabilistic neural network classification algorithm

Output: Bagging classifier $C(X)$

(a) . For i=1 to T

(b). n samples are extracted by a randomly put back extraction from the training set D . Create a sample set D_i .

(c). Using Sample sets and probabilistic neural network classification algorithm A to train, probabilistic neural network classifier $c_i(x)$ is obtained.

(d) . End For

(e). Output bagging classifier $C(X) = \arg\max \sum_{i}^{T} c_i(x)$.

And then, the bagging classifier $C(X)$ is used to classify the unknown sample x .

1. When bagging classifier classifies the unknown sample x , every classifier $c_i(x)$ will get a classification result. Then T classifier votes, the most votes of the class is the classification result of the unknown sample x .
2. Output classification results by voting $C(x) = \arg\max \sum_{i}^{T} c_i(x)$.

4 Gene Expression Profiles Classification Method Based on Neighborhood Rough Set and Probabilistic Neural Networks Ensemble

In this paper, we propose an improved method based on neighborhood rough sets and probabilistic neural network ensemble classifier for cancer classification. The detailed steps of the ensemble classification method are as follows:

Step 1 Sorting all the genes using the algorithm Relief, and then we select the first n genes to constitute the initial feature gene subset G_n ;

Step 2 The algorithm FARNem based on neighborhood rough sets is used to reduce genes of feature genes subset G_n . Further, it will select the feature gene subset G_m which is appropriate to classification.

Step 3 Feature gene subset G_m is the input of the ensemble classification algorithm Bagging-PNN which is based on probabilistic neural network. And it will carry classification training on the input sample set of cancer gene expression profiles. After training, we get the classification model $C(X)$;

Step 4 Assessing classification model by using test samples;

5 Experimental Results and Analysis

Experimental datasets include Colon dataset[14], Gastric dataset[15]and Ovarian dataset[16]. Their cancer gene expression profiles are standard datasets to verify cancer classification and gene expression analysis method. Colon dataset contains a total of 62 samples, which include 22 normal samples and 40 abnormal samples. Each sample contains 2000 level values of gene expression. Ovarian dataset has a total of

253 samples, which include 91 normal samples and 162 abnormal samples. Each sample contains 15154 level values of gene expression. In our experiment, we use intestinal type gastric cancer data for Gastric dataset. It contains a total of 40 samples, which include 20 normal samples and 20 abnormal samples. Each sample contains 1519 level values of gene expression. Their main characteristics are shown in Table 1.

Table 1. Characterized description of the data sets

Gene expression profile	Gene	Sample	Class1	Class2
Colon dataset	2000	62	40(cancer)	22(normal)
Gastric dataset	1519	40	20(cancer)	20(normal)
Ovarian dataset	15154	253	162(cancer)	91(normal)

We use leave-one-out cross validation method. From the results of the Table 2, the proposed Bagging-PNN ensemble classifier can achieve 90.32% accuracy in the colon cancer dataset. In [9],the method can achieve 77.4% and 88.7% classification accuracy respectively. For ovarian cancer dataset, Bagging-PNN classifier can achieve 96.44% accuracy, which outperforms the boosting method in [9] with 7.14% increasing. In gastric cancer dataset, the [10] used genetic algorithm to select 28 information genes which classification accuracy is 100%. But the proposed algorithm only needs 3 characteristic genes to get 100% classification accuracy.

Table 2. The classification results of the data sets

Gene expression profile	The number of feature genes	Correctly identify the number of samples
Colon dataset	3	56/62
Gastric dataset	3	40/40
Ovarian dataset	9	244/253

6 Conclusion

In this paper, we propose a new classification method based on cancer gene expression profiles structural characteristics. Firstly, we propose feature gene selection method based on neighborhood rough set to select a handful of feature genes with strong classification ability from thousands of genes. And then, we propose an ensemble classification algorithm based on the Bagging-PNN. Classification experiments on three gene expression datasets using this method have achieved better results than the method used in [9] and [10]. Experimental results show that the method we propose has achieved higher classification accuracy and stability. It is effective and practical for cancer gene expression data classification.

References

1. Quackenbush, J.: Microarray Analysis and Tumor Classification. The New England Journal of Medicine 354(23), 2463–2472 (2006)
2. Levi, D., Ullman, S.: Learning to Classify by Ongoing Feature Selection. Image and Vision Computing 28(4), 715–723 (2010)
3. Lee, Z.-J.: An Integrated Algorithm for Gene Selection and Classification Applied to Microarray Data of Ovarian Cancer. Artificial Intelligence in Medicine 42, 81–93 (2008)
4. Moon, H., Ahn, H., Kodell, R.L., Back, S., Lin, C.-J.: Ensemble Methods for Classification of Patients for Personalized Medicine with High-dimensional Data. Artificial Intelligence in Medicine 41, 197–207 (2007)
5. Zhang, L.-J., Li, Z.-J.: Gene Selection for Cancer Classification in Microarray Data. Chinese Journal of Computer Research and Development 46(5), 794–802 (2009)
6. Yeh, J.-Y.: Applying Data Mining Techniques for Cancer Classification on Gene Expression Data. Cybernetics and Systems: An International Journal 39, 583–602 (2008)
7. Hansen, L.K., Salamon, P.: Neural Network Ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(10), 993–1001 (1990)
8. Tan, A.C., Gilbert, D.: Ensemble Machine Learning on Gene Expression Data for Cancer Classification. Applied Bioinformatics 2(3), S75–S83 (2003)
9. Ben-Dor, A., Bruhn, L., Friedman, N., et al.: Tissue Classification with Gene Expression Profiles. In: Proceedings of the Fourth Annual International Conference on Computational Molecular Biology, vol. 5, pp. 1–32 (2000)
10. Li, H., Wang, J.-L.: Study of Tumor Molecular Prediction Model based Gene Expression Profiles. Acta Electronica Sinica 36(5), 989–992 (2008)
11. Kira, K., Rendell, L.A.: The Feature Selection Problem: Traditional Methods and a New Algorithm. In: Swartout, W. (ed.) Proceedings of the Tenth National Conference on Artificial Intelligence, pp. 129–134. AAAI Press, The MIT Press, San Jose, Cambridge (1992)
12. Hu, Q., Yu, D., Xie, Z.: Neighborhood Classifiers. Expert Systems with Applications 34, 866–876 (2008)
13. Specht, D.F.: Probabilistic Neural Networks. Neural Networks 3, 109–118 (1990)
14. Alon, U., Barkai, N., Notterman, D.A., et al.: Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays. Cell Biology 96, 6745–6750 (1999)
15. Boussioutas, A., Li, H., Liu, J., et al.: Distinctive Patterns of Gene Expression in Premalignant Gastric Cancer. Cancer Research 63, 2569–2577 (2003)
16. Petricoin, E., Ardekani, A., Hitt, B., et al.: Use of Proteomic Patterns in Serum to Identify Ovarian Cancer. Lancet 359, 572–577 (2002)

A New Hybrid Approach for Medical Image Intelligent Classifying Using Improved Wavelet Neural Network*

Jiang Yun, Xie Guocheng, Chen Na, and Chen Shan

College of Computer Science and Engineering, Northwest Normal University, 730070, Lanzhou Gansu, P.R. China
{jiangyun,chengguoxie,chenna,chenshan}@nwnu.edu.cn

Abstract. Breast cancer is the second leading reason of fatality among all cancers for women. In this paper, we propose a novel method for early breast cancer intelligent classification. We combine wavelet theory with neural network theory to construct an improved wavelet neural network (IWNN) for digital mammography classification. Firstly, we combine redundant dyadic wavelet transform with ridgelet transform to enhance the image. Because most of the wavelet coefficients containing signals are retained, the image detail can be kept better. And then, the statistical coefficients of the source regions are extracted as features for classification. At last, the medical images are classified by using IWNN on real datasets MIAS(the Mammographic Image Analysis Society). The experimental results show that proposed IWNN classifier can achieve 86.71% accuracy, which outperform the traditional neural network method with 5.46% of increase of classification accuracy. The correct recognition rates are close to 100% averagely.

Keywords: mammography, wavelet transform, ridgelet Transform, Neural Networks.

1 Introduction

Breast Cancer is the second leading cause of death in women. At present, lacking of effective technological aides makes higher misdiagnosis rate[1-3]. In [2], the authors use data mining techniques to classify anomalies in the breast. In [4], the authors use a Bayesian network to find and classify regions of interesting. In [5], the authors present a method for feature extraction of lesions in mammograms. In [6], the authors find an evaluation of different methods that can be used to get texture features from regions of interest extracted from mammogram images. In [7], we can see how the rough set has

* This work was partially supported by the National Natural Science Foundation of China (NSFC) under grant No. 61163036, No.61263036. The Natural Science Foundation of Gansu under grant No.1010RJZA022 and No. 1107RJZA112. The Fundamental Research Funds for Universities of Gansu Province. The Foundation of Gansu University Graduate Tutor No.1201-16. The third Knowledge Innovation Project of Northwest Normal University No.nwnu-kjcxgc-03-67.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 490–495, 2013.

been used to detect groups of micro-calcifications in digital mammograms. Ribeiro et al. [8] use text features and association rules to classify mammogram images. In addition, some other methods were presented in the literature[9-11]. With all this effort, there is still no widely used method to classify medical images.

Artificial neural network has great application effect to the knowledge that lacks relation between attribute and class. At present, many articles put forward its improved algorithm[12-14], but each of the algorithms has limitations in the application. In this paper, we proposed a novel method for mammograms classification. Firstly, we preprocess the image by combining redundant dyadic wavelet transform with ridgelet transform to enhance the image. And then, we propose an improved wavelet neural network (IWNN), which makes full use of wavelet transform, and analyzes image signal using multi-scale through dilation and shift operations. As a consequence, the IWNN has stronger capacity of approximation and fault tolerance which can classify the breast cancer data more efficiently.

2 Image Classification Based on Improved Wavelet Neural Network Algorithm

2.1 Image Preprocessing

Firstly, the image signal is decomposed by using wavelet through the soft threshold formula $\eta(\omega) = (\omega - \mathrm{sgn}(\omega)\mathrm{T})\mathrm{I}(|\omega| > \mathrm{T})$ Where, is the wavelet transform coefficient, T is pre-selected threshold. Then, the high frequency coefficient gets thresholds quantization. Finally, the image signal is reconstructed using two-dimensional wavelet. For the medical image $I(m,n)$, after L layer of wavelet transform, original image $I(m,n)$ can be decomposed into a low frequency sub-image $I_L(m,n)$ and a series of sub-band images $\{W_j^d(m,n)\}_{j=1,2...,L;d=1,2}$. According to the sizes of the two components, position and attribute of the edge can be judged. Enhance processing is to transform these sub-band images $\{W_j^d(m,n)\}_{j=1,2...,L;d=1,2}$. $\widehat{W}_j^d(m,n) = f_j^d\left(W_j^d(m,n)\right)$ where $f_j^d, d=1,2$ is the enhance function in scale *j*. We get the image from $\left\{\widehat{W}_j^d(m,n)\right\}_{j=1,...L;d=1,2}$'s reconstruction to processed images. Then the images reconstructed are transformed using *j* layer of the two-dimensional wavelet transform. With formula (1) and after ridgelet transform, the coefficient is x, and the coefficient after enhance transform is y.The function $f(x,y)$ is dyadic wavelet transform. While strengthening, noises can not be amplified and not distorted in the original clear edge, only coefficients in $[T_{\min}, T_{\max}]$ are corrected. Coefficients out of $[T_{\min}, T_{\max}]$ remain invariant, where p decides nonlinear of the enhancement curve. $T_{\min} = c\sigma$, where σ is the noise standard deviation of original image, and c is

the parameters to be determined. Here we suppose $c = 3$. $T_{\max}$ can be determined on the basis of the noise standard deviation.

$$\begin{aligned}\begin{pmatrix} W_j^1 f(x,y) \\ W_j^2 f(x,y) \end{pmatrix} &= 2^j \begin{pmatrix} \left(f^* \psi_j^1\right)(x,y) \\ \left(f^* \psi_j^2\right)(x,y) \end{pmatrix} \\ &= 2^j \begin{pmatrix} \frac{\partial}{\partial x}\left[\left(f^* \theta_j\right)(x,y)\right] \\ \frac{\partial}{\partial y}\left[\left(f^* \theta_j\right)(x,y)\right] \end{pmatrix} = 2^j \nabla \left(f^* \theta_j\right)(x,y)\end{aligned} \tag{1}$$

After pre-processing the images, features relevant to the classification are extracted from the cleaned images. The extracted features are four kinds of statistical parameters: *mean*, *variance*, *partial gradient* and *kurtosis*[2].

2.2 Improved Wavelet Neural Network (IWNN)

In this paper, the wavelet function is embedded in neural network, which is called wavelet neurons instead of the traditional neurons in back-propagation neural network. And the wavelet function takes the place of the conventional neural network of hidden function. The weights of corresponding input layer to the hidden and the threshold of hidden layer are replaced by wavelet function's scale and translation parameters respectively. By means of wavelet theory, the neural network has a simple topology structure and higher convergence speed. But the scale and the translation parameters are adjustable, which make network and its output have nonlinear relationship. By using nonlinear optimization method to fix the parameters, it is easy to have minimum weakness in fix similar BP network parameters.

Let us suppose that the improved wavelet neural network have three layers. The neurons activation function is discrete dyadic wavelet:

$$\psi_{m,k}(x) = \sqrt{2^{-m}}\,\psi\left(2^{-m}x - k\right), \quad k,\ m \in \mathbf{Z}^2. \tag{2}$$

Where m, k is stretch and shift factor respectively. We introduce two filters *H* and *G*, which impulse responses are:

$$h_k = \int_{-\infty}^{+\infty} \varphi_{0k}(x)\varphi_{1k}(x)\,dx,\ g_k = \int_{-\infty}^{+\infty} \varphi_{0k}(x)\psi_{1k}(x)\,dx \tag{3}$$

Where $a_{1k} = \sum_{k=-\infty}^{\infty} h_k a_{0k}$, $d_{1k} = \sum_{k=-\infty}^{\infty} g_k a_{0k}$

For a group of discrete samples, we can get the following recursive formula:

$$A_{m-1}F(x) = \sum_{k\in\mathbb{Z}} a_{mk}\varphi_{mk}(x) + \sum_{k\in\mathbb{Z}} d_{mk}\psi_{mk}(x). \tag{4}$$

Where coefficient a_{mk}, d_{mk} satisfy:

$$a_{mk} = Ha_{m-1k}, d_{mk} = Ga_{m-1k}, \forall k. \tag{5}$$

This forms the wavelet network which has two units in hidden layer.

(1). Scale functions $\phi(x)$ unit $\phi_{Lk}(x)$, $k = 1, 2, \cdots, n_L$, are orthogonal basis functions in different displacement, when under the resolution L (lowest resolution), and then construct approach for unknown function in a minimum of the resolution.

(2). The wavelet function $\psi(x)$ and unit $\psi_{mk}(x)$ are the orthogonal basis functions of the details of $F(x)$.

The algorithm of improved wavelet neural network (IWNN) is as following:

Algorithm two: Improved Wavelet Neural Network (IWNN)

Input: Network parameters initialization (Wavelet's adjustable factor a_k, translation factor b_k, network connection weights w_{km} and w_{nk}, learning rate $\eta(\eta > 0)$ and momentum factor $\lambda(0 < \lambda < 1)$ are given initial value. And it input sample calculator $p = 1$).

Output: Classification results Y

(a) Input study sample and the corresponding expected output value d_n^p.

(b) With the fast wavelet transform the original signal into different scales.

(c) Calculate the output of hidden and output layer, and deposit in the prediction matrix.

(d) Calculate error and gradient vector;

(e) Input next sample $p = p + 1$;

(f) Judge if algorithms are over. When $E < \varepsilon$, cost function E less than pre-set precision value $\varepsilon(\varepsilon > 0)$, stop network learning; otherwise reset p for 1, and then return to (a).

3 Experimental Results

In our experiments, the data collection was taken from MIAS[2]. The first step is cutting operation on the image. The images are cropped to 454 × 605. Then, the wavelet packet of the images is decomposed to determine the optimal wavelet, and the wavelet packet decomposition coefficients are quantitative. Finally, the image wavelet packet is reconstructed into de-noised images. In this paper, the dyadic wavelet transform and ridge wavelet transform method are combined together to enhance the image. We extract 64 feature values for each image. And then, the initial values of the weight factor are set. The initial size is 15. The original units are two ϕ nodes. The ψ nodes are increased one by one for improving the approximation accuracy. The different levels of approximation error and prediction error are shown in Table 1 for the established wavelet network. At last, 12 is selected as the final size. After 100 times training, the Standard Deviation is 0.0742244/0, and Gradient is 0.0104811/1e-006. The error is smaller when the prediction classifier is used.

Table 1. Different levels of approximation error and prediction error with initial weight factor

No.	Size	New units	Approximation error	Prediction error
1	15	two ϕ	1.832	1.376
2	15	two ψ	0.532	0.343
3	14	three ψ	0.203	0.132
4	13	four ψ	0.133	0.156
5	12	five ψ	0.183	0.038

We use the 10 fold cross-validation techniques to evaluate the algorithm performance. There are 288 images in the training set and 34 images in the testing set. The results of the experiment are shown in Figure 1, in which the abscissa shows the data set's 10 times randomly partition, and the vertical axis shows classification accuracy of 10 times. The average classification accuracy using IWNN algorithm is 86.71 %, which is higher than [2].

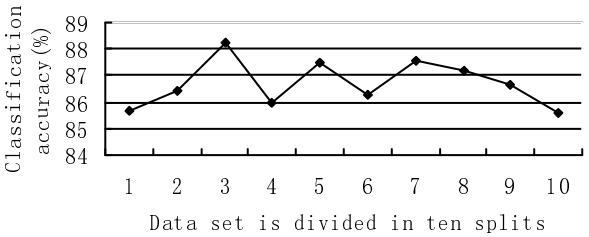


Fig. 1. Experimental results on the MIAS Dataset using IWNN algorithm

In our experiments, the positive predictive values and the negative predictive values are close to 100%. This shows that there is a high rate of correct classification on normal and abnormal mammography. It is expected by medical experts.

4 Conclusion

In this paper, we construct a novel improved wavelet neural network (IWNN) classifier for medical devices. We use the MIAS dataset to evaluate the performance of our method. Experiments show that the average classification accuracy by using this algorithm is 86.71%, which is higher about 5.46% than 81.25% in [2]. There is a very low probability to wrongly merge the abnormal images into others. It can be able to meet the needs of the actual medical diagnosis.

References

1. Shah, S., Kusiak, A.: Cancer Gene Search with Data-mining and Genetic Algorithms. Journal of Computers in Biology and Medicine 37(2), 251–261 (2007)
2. Antonie, M.L., Zaiane, O.R., Coman, A.: Application of Data Mining Techniques for Medical Image Classification. In: Proceeding of Second International Workshop on Multimedia Data Mining (MDM/KDD 2001) in Conjunction with Seventh ACM SIGKDD, San Francisco, USA, pp. 94–101 (2001)
3. Delen, D., Walker, G., Kadam, A.: Predicting Breast Cancer Survivability: A Comparison of Three Data Mining Methods. Journal of Artificial Intelligence in Medicine 34(2), 113–127 (2005)
4. Zhang, X.-P., Desai, M.D.: Wavelet Based Automatic Thresholding for Image Segmentation. In: Proceedings of the ICIP 1997 Conference, Santa Barbara, CA, pp. 26–29 (1997)
5. Bottigli, U., Golosio, B.: Feature Extraction from Mammographic Images Using Fast Matching Methods. Journal of Nuclear Instruments and Methods in Physics Research 487(A), 209–215 (2002)
6. Sharma, M., Singh, S.: Evaluation of Texture Methods for Image Analysis. In: Proceedings of the 7th Australian and New Zeland Intelligent Information Systems Conference, Perth, pp. 18–21 (November 2001)
7. Ilczuk, G., Wakulicz-Deja, A.: Rough Sets Approach to Medical Diagnosis System. In: Szczepaniak, P.S., Kacprzyk, J., Niewiadomski, A. (eds.) AWIC 2005. LNCS (LNAI), vol. 3528, pp. 204–210. Springer, Heidelberg (2005)
8. Ribeiro, M.X., Traina, A.J.M., Balan, A.G.R., Traina Jr., C., Marques, P.M.A.: SuGAR: A Framework to Support Mammogram Diagnosis. In: Proceeding of IEEE CBMS, Slovenia, Maribor, pp. 47–52 (2007)
9. Sarkar, M.: Fuzzy-rough Nearest Neighber Algorithms in Classification. Journal of Fuzzy Sets and Systems 158(19), 2134–2152 (2007)
10. Li, H., et al.: Markov Random Field for Tumor Detection in Digital Mammography. IEEE Trans. Medical Imaging 14(3), 565–576 (2000)
11. Mantzaris, D., Anastassopoulos, G., Adamopoulos, A.: Genetic Algorithm Pruning of Probabilistic Neural Networks in Medical Disease Estimation. Journal of Neural Networks 24(8), 831–835 (2011)
12. Jain, L.C.: Advances in Design and Application of Neural Networks. Journal of Neural Computing & Applications 19(2), 167–168 (2010)
13. David, J., Krishnan, R., Kumar, S.: A Neural Network based Retinal Image Analysis. In: Proceedings of the 2008 Congress on Image and Signal Processing, pp. 49–53 (2008)
14. Ramírez, J., Chaves, R., Górriz, J.M., et al.: Functional Brain Image Classification Techniques for Early Alzheimer Disease Diagnosis. Journal of Bioinspired Applications in Artificial and Natural Computation 56(02), 150–157 (2009)

Enhancing SOM Based Visualization Methods for Better Data Navigation

Ying Wang (Florence) and Masahiro Takatuska

Vislab, School of IT, The University of Sydney
Sydney, Australia
{florence8627,masa}@vislab.net

Abstract. Existing SOM based visualization methods have the limitation of not being able to show detailed local distance information and global similarity of data at the same time. In this paper, we propose two approaches to overcome this limitation for better data navigation. In our experiments, we used MPEG-7 shape image dataset and classic IRIS dataset to demonstrate our approaches are superior to previous approaches in providing sufficient local and global information for data visual navigation.

Keywords: Self Organizing Map(SOM), Data visualization and navigation, Local distances preserving, Global distances preserving.

1 Introduction

Self Organizing Map (SOM) [1] is widely used in exploratory data analysis and visualization. Through unsupervised training, SOM can project high-dimension data to low-dimension (typically 2) regular grids which is called the "map". The visualization and navigation of high-dimension data are made possible through their Best Matching Units (BMUs) [1] on the map. In the past, many SOM based methods were proposed for visualizing data cluster structure [2]. They can be generally categorized as local or global distance preserving techniques.

Among local distance preserving techniques, the most widely used are distance matrices [2]. U-matrix [3], for instance, calculates the distances of each map unit to its immediate neighbors, which can be visualized via gray shades. Usually darker shades indicate greater distances. A simplified version is to calculate a single value for each map unit (e.g. the averaged distances to its neighbors) and use it to control the size or color of the map unit for visualization [4]. Apart from distance matrices, Polzlbauer et al. [5] borrowed ideas from flow and vector field visualization, Tasdemir [6] borrowed ideas from graph visualization for visualizing local similarity of SOM map units.

Among global distance preserving techniques, one of the earliest is similarity color coding [7]. In this approach, a nonlinear projection is used to map the cluster structure of trained SOM to different hue values. As a result, the perceptual similarity between hues reflects the global order (i.e. similarity) between

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 496–503, 2013.

clusters. On the other hand, the perceived differences between hues reveals cluster borders. Himberg [8] introduced a contraction model to pull similar map units together and assign them with similar coloring iteratively to visualize the hierarchical clustering structure of data. Recently, Runz et al. [9] introduced a data-driven color mapping algorithm to visualize the global similarity of data cluster. The common problem in these previous approaches is that nodes within the same cluster tend to have uniform colors, hence detailed local distance information is lost.

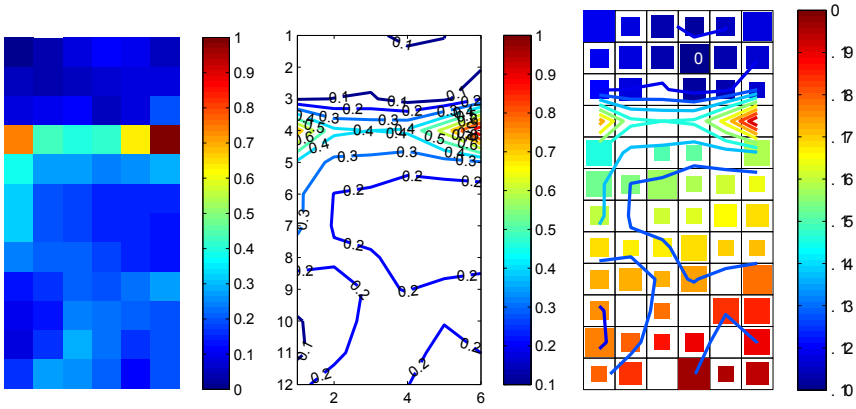
In summary, local distance preserving techniques help to visualize local similarity of each map unit to its neighbors. With local distance preserving visualizations [3–6], although local information such as local minima (i.e. cluster center) and cluster border can be easily identified, global relationships of separate clusters can not be captured. On the other hand, global distance preserving techniques help to visualize global order (i.e. similarity) of individual data clusters and identify possible distortions caused by dimension reduction (when two similar clusters are separately located on SOM). With global distance preserving visualizations [7–9], although global relationship of data clusters can be perceived, local information are not well preserved [7]: only the most important local information (i.e. cluster borders) are captured, information such as neighborhood distances and cluster centers are not discernible. Therefore, in this paper we propose two approaches to overcome the limitations of existing SOM based visualization methods. Using our proposed approaches, both local distances and global ordering of the data can be visualized on a single map to facilitate SOM based data navigation.

2 Our Approaches

In this section, two approaches are proposed for displaying both local distances and global ordering of input data on a single map. Classic IRIS dataset [10] which presents 3 different classes of iris flowers is used for illustrating our approaches. In IRIS dataset, 150 instances are defined by 4 variables.

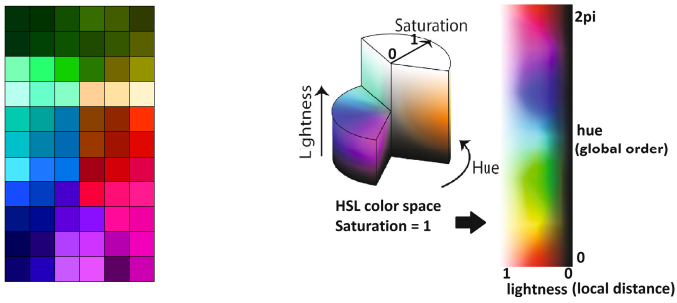
2.1 Contour Overlay with Interactive Coloring (1st Approach)

The idea of this approach is to use contour plot of distance matrix for displaying local distances, and colors of map units for interactively displaying the similarity between the user selected map unit and the rest of units. Such similarity can be pre-calculated through pair wised distances of model vectors. Fig. 1 illustrates this approach. In the distance matrix (Fig. 1a), small local distances (shown by deep blue) indicate cluster structures of the data, yet global similarity among clusters is not perceivable. Therefore, in Fig. 1c, we interactively re-color all the nodes based on their distances to user's current selection (node marked with "1") to show the global distances among all the nodes while user navigating



(a) Distance matrix (b) Contour plot (c) Our 1st approach

Fig. 1. Our 1st approach using IRIS dataset. (b) is the contour plot of (a). (c) is the visualization result: contour plot is imposed on SOM to show local distances, colors of nodes indicate the distances of every nodes to the user selected node (marked as "1"). Sizes of nodes denote the number of data hits.



(a) Our 2nd approach (b) Color legend for (a)

Fig. 2. Our 2nd approach using IRIS dataset. (a) is the visualization result: lightness of nodes indicate local distances: the smaller (bigger) the local distance, the darker (lighter) the node; similarity of hues indicate global similarity. (b) is the color legend which is obtained by unfolding HSL cylindrical surface.

through the map, meanwhile, local distances can be visualized via the contour overlay (Fig. 1b) imposed on the map. Sizes of the nodes in Fig. 1c indicate the number of data hits. The larger the size, the more data entries are mapped to the node. The algorithm of this approach is summarized as follows:

Contour Overlay with Interactive Coloring (1st Approach)

```
PROGRAM InteractiveApproach (InputData)
   TrainedSOM = TrainSOM(InputData);
   Calculate DistanceMatrix via Umatrix;
```

```
   Obtain ContourPlot of DistanceMatrix;
   Calculate PairWisedDistances of ModelVectors in TrainedSOM;
   Display DistaceMatrix with ContourPlot overlay;
   Allow user's selection;
   WHILE UserClick <= MaximumClick
          ClickedNode = IOHandling(UserClick);
          Distances =  PairWisedDistance(ClickedNode, OtherNodes);
          Display Distances with ContourPlot overlay;
   END WHILE
END PROGRAM
```

2.2 HSL Color space Mapping Based Color Coding (2nd Approach)

In the past, only single channel (typically hue) of a color model is chosen to show global similarity. Even if multiple channels are used, only one channel is used predominantly, other channels are used for smoothing color transitions. Therefore, using previous global distance preserving techniques, nodes within the same cluster tend to have uniform colors (see Fig. 5c). Although cluster borders are still perceivable, detailed local distance information (e.g. neighborhood distances and local minima) is lost. This will be further discussed in section 3.2.

To be able to visualize both local and global distances on the same map, we use hue and lightness in HSL color space, so that cluster border and global distances can be seen via hues, neighborhood distances and local minima can be seen via lightnesses. For instance, in Fig. 2a, the perceptual similarity of hues indicate global similarity of clusters; meanwhile, lightness of each node indicates neighborhood distances: the smaller (bigger) the local distance, the darker (lighter) the node. The range of colors used in our visualization is obtained by unfolding the HSL cylindrical surface at maximum saturation (Fig. 2b). After the initial SOM is trained with input data, hue and lightness of each node can be decided by training another SOM to map its model vectors to the HSL cylindrical space at maximum saturation (Fig. 2b). Here, we use two 1D SOMs trained with the model vectors and distance matrix to simplify this process, which can be easily extended to its full version by using a 2D SOM and possibly color interpolation. The algorithm of this approach is summarized as follows:

HSL Color space Mapping Based Color Coding (2nd Approach)

```
PROGRAM HSLMappingApproach (InputData)
   TrainedSOM = TrainSOM(InputData);
   Calculate DistanceMatrix via Umatrix;
   Train a 1D SOM  with 360 neurons ring topology
          to map ModelVectors of TrainedSOM to Hue channel;
   Train a 1D SOM  with 100 neurons chain topology
          to map DistanceMatrix to Lightness channel;
   Set Saturation to maximum value;
   HSLNodeColors = (Hue, Saturation, Lightness);
   RGBNodeColors = HSLToRGB(HSLNodeColors);
```

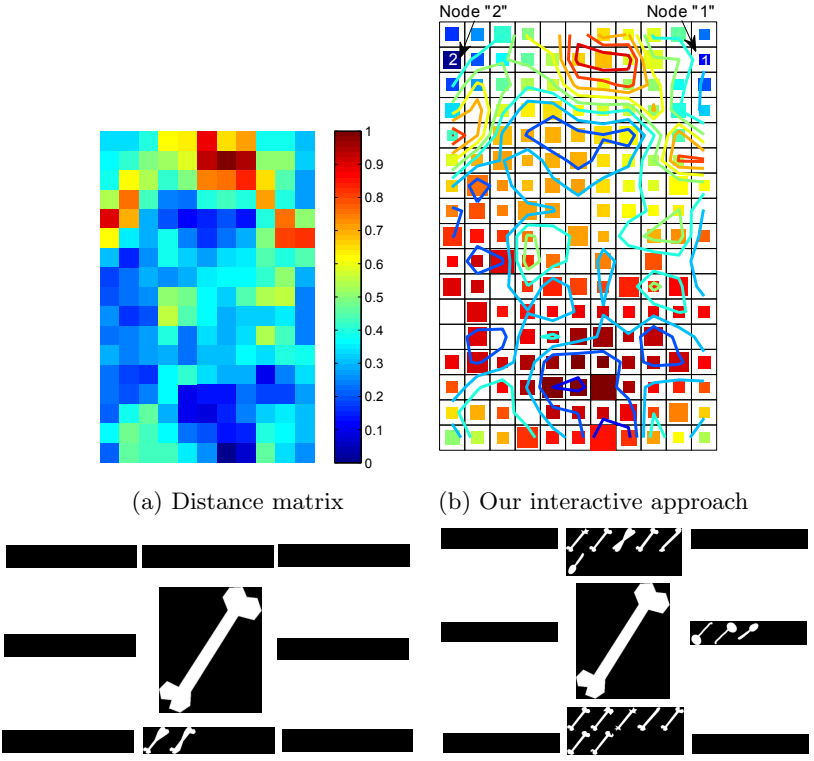
```
   Display SOM grids with RGBNodeColors;
END PROGRAM
```

3 Experimental Results

Our approaches are implemented in MATLAB using SOM toolbox [2]. MPEG-7 shape [11] image dataset and IRIS dataset [10] are used for our experiments.

3.1 Data Navigation of MPEG-7 Shape Image Dataset

MPEG-7 shape image dataset [11] (1400 images of 70 different shapes) is used for testing our proposed visualization approaches for data navigation. Space filling curve based shape descriptor [12] is applied to obtain our input data. The input data size is 1400×64. We use a 17×11 SOM with toroid topology to train the input data, the visualization results can be seen in Fig. 3 and Fig. 4.



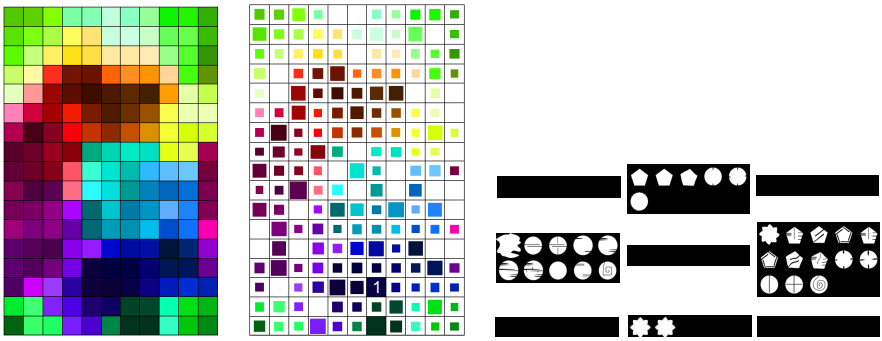
(a) Distance matrix (b) Our interactive approach

(c) Images mapped to node "1" in (b) (d) Images mapped to node "2" in (b)

Fig. 3. Data navigation of MPEG-7 dataset using our 1st approach. (a)(b) share the same color bar. In (b), colors of contour lines indicate local distances; colors of nodes indicate global distances relative to user's current selection (node "2").

In Fig. 3, the image ("bone.jpg") displayed in the center of Fig. 3c and Fig. 3d is the searched image while the user is navigating through the map. All the images around the center are the ones mapped to the clicked node. They are organized into eight positions based on which (1 out of 8) mediate neighbors of the clicked node they are closest to. Such organization of mapped images provides guidance of direction to the user when searching an image. For instance, the user can move the mouse in the same direction where similar images to the searched image are arranged. Also the user can also move the mouse based on the colors of map nodes: blue colors indicate similar images are mapped to them (Fig. 3c and Fig. 3d). Meanwhile, blue contour lines in Fig. 3b show detailed local distance information (i.e. cluster structures) within the dataset. Fig. 4 is the visualization result using our HSL colorspace mapping based method. Lightness of nodes show local distances (the darker (lighter) the node, the smaller (bigger) the averaged neighborhood distances); similarity of hues indicate global similarity among clusters. Therefore, in Fig. 4a, similar clusters are colored by similar hues although they may be separately located (e.g. the four corners of the map). Cluster borders can be seen via different hues, which also appear lighter. Darkest nodes within clusters denote local minima. Fig. 4b also enables users to navigate the data by clicking the map nodes (similar to Fig. 3c, 3d). Fig. 4c shows such an example.

In summary, either the interactive approach (Fig. 3) or the color space mapping approach (Fig. 4) is able to display sufficient local and global information of data at the same time. Particularly, by interactively displaying mapped image data of each map node, our approaches facilitate the visual navigation of data.

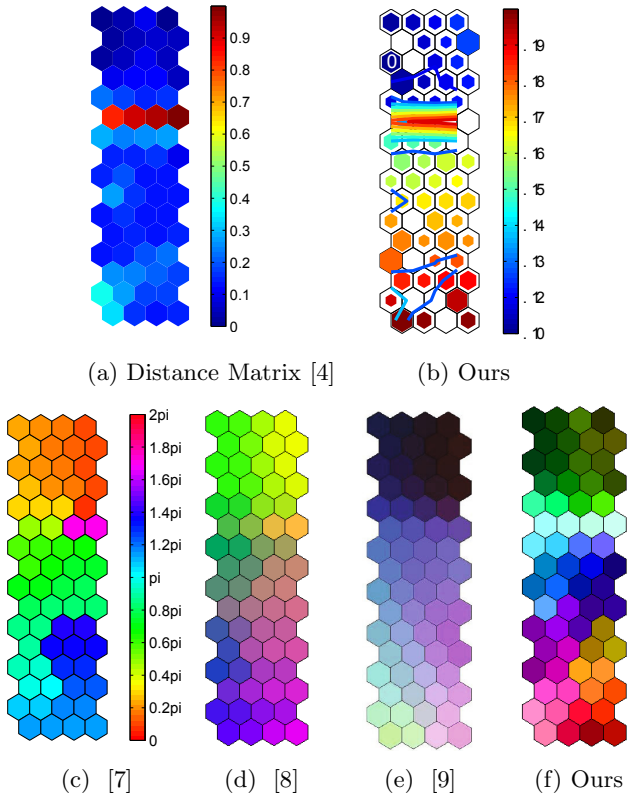


(a) Our colorspace mapping approach

(b) Data hits shown by sizes of nodes

(c) Image data mapped to node marked as "1" in (b)

Fig. 4. Data navigation of MPEG-7 dataset using our 2nd approach. In (a)(b), the smaller (bigger) the local distance, the darker (lighter) the node; similarity of hues indicate global similarity of clusters. In (b), sizes of nodes denote how many data entries (images) are mapped to the node.



(a) Distance Matrix [4] (b) Ours

(c) [7] (d) [8] (e) [9] (f) Ours

Fig. 5. Comparison of our approaches to previous approaches using IRIS dataset. (a)(c)(d)(e) are introduced by [4] [7] [8] [9]. Perceptual differences of hues indicate dis-similarities of nodes. (b) is our interactive method: colors of contours indicate local distances; colors of nodes indicate global distances relative to user's selection (marked as "1"). (f) is our HSL colorspace mapping method. See Fig 2b for color legend of (f).

3.2 Visualization Methods Comparison Using IRIS Dataset

We compare the visualization results of our approaches with previous approaches using IRIS dataset [10] which represents 3 classes of iris flowers having size of 150×4. In Fig. 5, we train a 16×4 SOM with hexagon lattice to generate visualization results using our approaches and global distance preserving techniques introduced by [4, 7, 8]. Fig. 5e is extracted from [9]. In the distance matrix [4] (Fig. 5a), although local distances, cluster borders are shown; global distances of clusters are not discernible. In Fig. 5c, 5d and 5e, although similarity of clusters and cluster borders are perceivable, colors within the same cluster tend to be uniform, hence detailed local information such as neighborhood distances and local minima can not be visualized. However, both our approaches have overcome above limitations and are able to display sufficient local information and global similarity of data clusters at the same time to facilitate better data navigation.

4 Conclusion

In this paper, we proposed two approaches to enhance existing SOM based data visualization methods for better data navigation. Using our approaches, both detailed local information and global ordering of the data can be visualized on the same map. Through our experiments, we have shown our approaches have overcome the limitations of previous approaches and facilitated SOM based data navigation.

References

1. Kohonen, T.: The self-organizing map. Proceedings of the IEEE 78(9), 1464–1480 (1990)
2. Vesanto, J.: Som-based data visualization methods. Intelligent Data Analysis 3(2), 111–126 (1999)
3. Ultsch, A., Siemon, H.P.: Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis. In: Proceedings of International Neural Networks Conference (INNC), Paris, pp. 305–308 (1990)
4. Kraaijveld, M.: A non-linear projection method based on kohonen's topology preserving maps. In: Proceedings of the 11th IAPR International Conference on Pattern Recognition Methodology and Systems, vol. 2, pp. 41–45 (1992)
5. Polzlbauer, G., Dittenbach, M., Rauber, A.: A visualization technique for self-organizing maps with vector fields to obtain the cluster structure at desired levels of detail. In: International Joint Conference on Neural Networks (IJCNN), vol. 3, pp. 1558–1563 (2005)
6. Tasdemir, K.: Graph based representations of density distribution and distances for self-organizing maps. IEEE Transactions on Neural Networks 21(3), 520–526 (2010)
7. Kaski, S., Venna, J., Kohonen, T.: Coloring that reveals high-dimensional structures in data. In: 6th International Conference on Neural Information Processing (ICONIP), vol. 2, pp. 729–734 (1999)
8. Himberg, J.: A som based cluster visualization and its application for false coloring. In: International Joint Conference on Neural Networks (IJCNN), pp. 587–592 (2000)
9. De Runz, C., Desjardin, E., Herbin, M.: Unsupervised visual data mining using self-organizing maps and a data-driven color mapping. In: 16th International Conference on Information Visualisation (IV), pp. 241–245 (2012)
10. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annals of Eugenics 7(2), 179–188 (1936)
11. MPEG-7: Shape dataset, `http://www.cis.temple.edu/~latecki/TestData/mpeg7shapeB.tar.gz`
12. Ebrahim, Y., Ahmed, M., Chau, S., Abdelsalam, W.: An efficient shape representation and description technique. In: IEEE International Conference on Image Processing, ICIP 2007, vol. 6, pp. 441–444 (2007)

Classification of Physiological Sensor Signals Using Artificial Neural Networks

Nandita Sharma and Tom Gedeon

Research School of Computer Science
Australian National University, Canberra, Australia
{nandita.sharma,tom.gedeon}@anu.edu.au

Abstract. Physiological signals have certain prominent characteristics that distinguish them from other types of physiological signals which are familiar to experts and assessed by inspection. The aim of this paper is to develop a computational model that can distinguish electrocardiogram, galvanic skin response and blood pressure signals acquired from sensors as well as detect corrupted signals which can arise due to hardware problems including sensor malfunction. Our work also investigates the impact of the signal modeling for various time lengths and determines an optimal signal time length for classification. This provides a method for automatic detection of corrupted signals during signal data collection which can be incorporated as a support tool during real-time sensor data acquisition.

Keywords: signal classification, artificial neural networks, physiological signals, time series data, signal modeling.

1 Introduction

Physiological signals are generated by the human body and have been analyzed to classify different states of a person including health condition detection [1-3] and affective state classification [4, 5] however little attention has been given to develop models for model free recognition of physiological signals and detection of corrupted signals. Filtering techniques have been utilized for artifact classification in physiological signals such as EEG signals [6]. Our work is focused on computationally capturing the underlying properties that distinguish the nature of the different types of signals and separate the different types of signals.

Artificial neural networks (ANNs), inspired by biological neural networks, have characteristics for learning patterns to classify input tuples into classes. It is made up of interconnected processors, known as *artificial neurons*, which are connected by weighted links that pass signals between neurons to learn relationships between tuples and output classes. In this work, we used feed-forward ANNs trained using backpropagation to generate signal classification models.

This paper presents the signal data that will be modeled using ANNs for classification. The ANN models to model and classify physiological signals and corrupted signals using individual-independent models and models for a particular individual are proposed. We provide results of the ANNs on the data and analyses of the results.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 504–511, 2013.

We also investigated how the length of the signal affects the performances of the ANNs for signal classification. The paper concludes by summarizing the work and suggests future work.

2 Physiological Signal Sensor Data

The physiological sensor signal data used for our models were obtained from the data set collected in [7]. Three different types of physiological signals are used in this work and they are electrocardiogram (ECG), blood pressure (BP) and galvanic skin response (GSR). Examples of the signals in the data set are shown in Fig. 1.

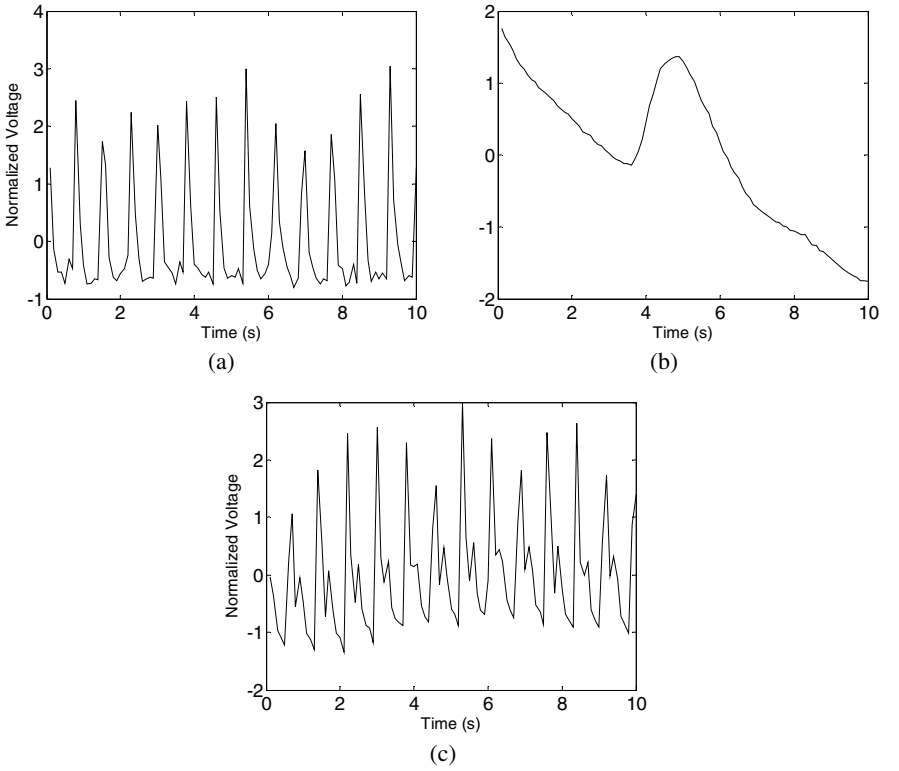


Fig. 1. Sample physiological signals (a) ECG signal (b) GSR signal (c) BP signal

The physiological signals modeled in this work are produced by different activities in the Autonomic Nervous System of the human body. An ECG signal captures electrical activity produced by the impulse of ions flowing through cardiac muscles, which dissipates into the region around the heart with diminished amounts spreading around the surface of the body. The ECG waveform is characterized by the dominant QRS wave where the R is the peak of the wave. ECG signals can be used to determine cardiovascular fitness, and dynamic and cumulative load of a person [8].

A GSR signal provides a measurement of the flow of electricity through the skin of an individual. Variations in GSR have been found to reflect stress levels in

individuals while they played a competitive racing game [9]. In addition, cognitive load [10] and work performance [11], which can be seen as stressors [12, 13], have strong correlations with GSR. GSR waveforms may have consistent shapes in reaction to stressors but are not usually periodic.

A BP signal shows the amount of pressure exerted on the walls of blood vessels due to blood circulation. The signal shows variations of the pressure between a systolic (maximum) and a diastolic (minimum) pressure.

The physiological sensor signal data set was used to model the three types of signals spanning 15 minutes for 22 subjects. For the purpose of this work, the signals were sampled at 10 Hz and this captured the main properties of the physiological signals such as the QRS waveforms in the ECG signals and the systolic and diastolic pressures in the BP signals as shown in Fig. 1. There were 10 other subjects who had their signals recorded but at least one of their signals were corrupted by manual inspection. This data was used to develop modeling systems that recognized corrupted signals as well as the physiological signals. Further, the signals were normalized to minimize the impact of individual bias, offset and noise in the signals for modeling and to better capture the underlying properties of the signals such as the QRS waveform for ECG signals.

3 Artificial Neural Network Signal Classifiers

ANN models were developed to recognize the different physiological signals and corrupted signals. The ANNs differed in terms of the data modeled and the topology. They are described as follows:

1. **ANN-10s:** the ANN modeled signals segmented in 10 seconds time segments and used data from all subjects for training and testing the model
2. **ANN-Ind-10s:** the ANN modeled signals segmented in 10 seconds time segments and used data from a particular individual (i.e. one subject) for training and testing the model
3. **ANN-10s-Corrupt:** the ANN modeled signals segmented in 10 seconds time segments and used data from all subjects and subjects who had corrupted signals for training and testing the model

Similarly, **ANN-5s**, **ANN-Ind-5s**, **ANN-5s-Corrupt, ANN-1s**, **ANN-Ind-1s**, **ANN-1s-Corrupt, ANN-0.5s**, **ANN-Ind-0.5s** and **ANN-0.5s-Corrupt** were developed for signals segmented in 5 seconds, 1 second and 0.5 seconds time segments. The ANNs that modeled corrupted signals in addition to the physiological signals had four output neurons, which was one more neuron than the ANNs that did not model the corrupted signals.

Each type of ANN defined above had three different topologies for the hidden layers:

1. One hidden layer with 7 neurons
2. Two hidden layers with 7 neurons in the first hidden layer and 5 neurons in the second hidden layer
3. Three hidden layers with 7 neurons in the first hidden layer, 5 neurons in the second hidden layer and 3 neurons in the third hidden layer

Additionally, ANN models were developed that took two types of physiological signals as input:

1. ANN-ECG-GSR: the ANN was modeled to recognize ECG and GSR signals
2. ANN-ECG-BP: the ANN was modeled to recognize ECG and BP signals
3. ANN-GSR-BP: the ANN was modeled to recognize GSR and BP signals

All the ANNs were implemented and tested using MATLAB. The MATLAB *adapt* function was used for training the ANN on an incremental basis. Each ANN was trained using the Levenberg-Marquardt algorithm for 1000 epochs or until the magnitude of the gradient for the mean squared error (MSE) was less than 10^{-5} during the validation phase.

4 Results and Discussion

The ANNs for signal recognition were trained and tested on the sensor signal data sets collected in [7] using 10-fold cross-validation process. The process was executed 20 times to obtain the mean and standard deviation of the recognition rates for the different types of signals.

Results of the individual-independent ANNs for physiological signal classification are shown in Fig. 2. ANN-1s produced the best recognition rates for all the signals and the results were statistically significant according to the Student's T-test ($p < 0.001$).

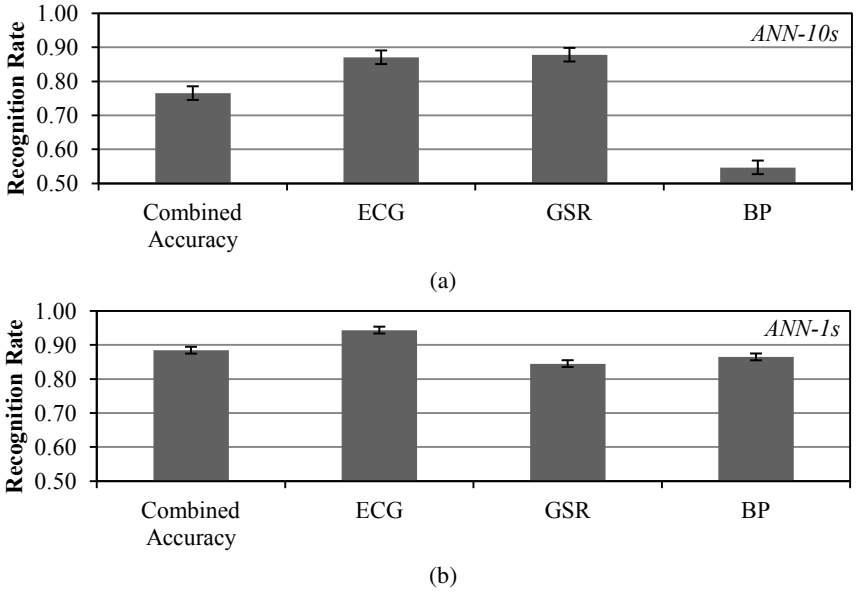
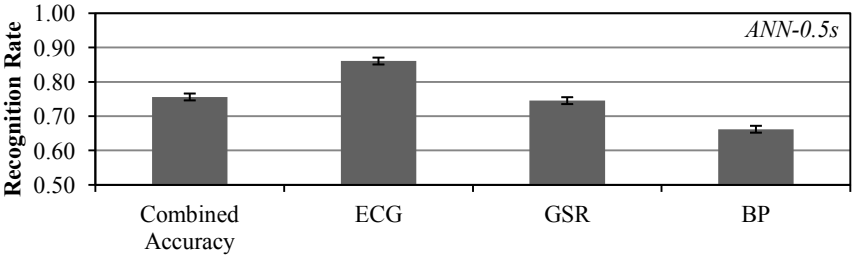


Fig. 2. Recognition rates for the physiological signals from individual-independent ANN classifiers based on 10-fold cross-validation (a) ANN-10s – its recognition rates were similar to ANN-5s (b) ANN-1s – it produced optimal results (c) ANN-0.5s



(c)

Fig. 2. (*Continued*)

The trend in the signal recognition rates for ANN-Ind-10s on each individual signal data set was statistically similar ($p < 0.05$) to the trend in the signal recognition rates for ANN-10s in that the GSR, ECG and BP signals had the highest, second highest and the lowest recognition rates respectively. The recognition rates for ANN-Ind-10s are provided in Fig. 3. Trends in the recognition rates for ANN-Ind-5s, ANN-Ind-1s and ANN-Ind-0.5s were similar to ANN-5s, ANN-1s and ANN-0.5s in the same way as well.

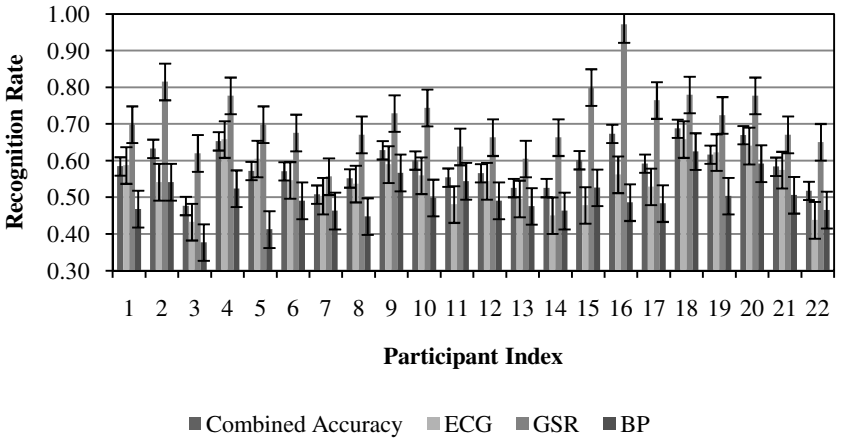


Fig. 3. Recognition rates for the physiological signals for individuals from ANN classifiers based on 10-fold cross-validation for ANN-Ind-10s

The recognition rates of the ANNs modeled on only two types of signals using the data provided to ANN-10s are provided in Table 1. The results show that ANN-ECG-BP produced the lowest classification rates compared to ANN-ECG-GSR and ANN-GSR-BP. The accuracy for ANN-ECG-BP was at least 0.27 lower than the other two ANNs. The ANN could not distinguish the ECG and BP signals as strongly as GSR

and the other types of signals. From the results, GSR signals were less similar to the other two types of signal so this explains why the GSR recognition rates were the highest for ANN-10s and with similar reasoning, it explains the trend in the recognition rates for the other types of signals. Further, the data provided to ANN-5s, ANN-1s and ANN-0.5s were provided to ANNs that modeled two types of signals to explain their trends in a similar fashion.

Table 1. Signal recognition rates produced from ANN models classifying two types of signals

ANN	Accuracy	ECG	GSR	BP
ANN-ECG-GSR	0.99	1.00	0.99	-
ANN-ECG-BP	0.68	0.73	-	0.62
ANN-GSR-BP	0.95	-	0.96	0.95

The recognition rates for ANN-10s-Corrupt, ANN-5s-Corrupt, ANN-1s-Corrupt and ANN-0.5s-Corrupt are shown in Fig. 4. Results show that the recognition rate for corrupted signals was the highest for ANN-10s-Corrupt compared to the other ANNs that recognized corrupted signals. Nevertheless, the ANN-1s-Corrupt produced the highest combined classification accuracy and the highest recognition rates for the other signals i.e. physiological signals just as ANN-1s did.

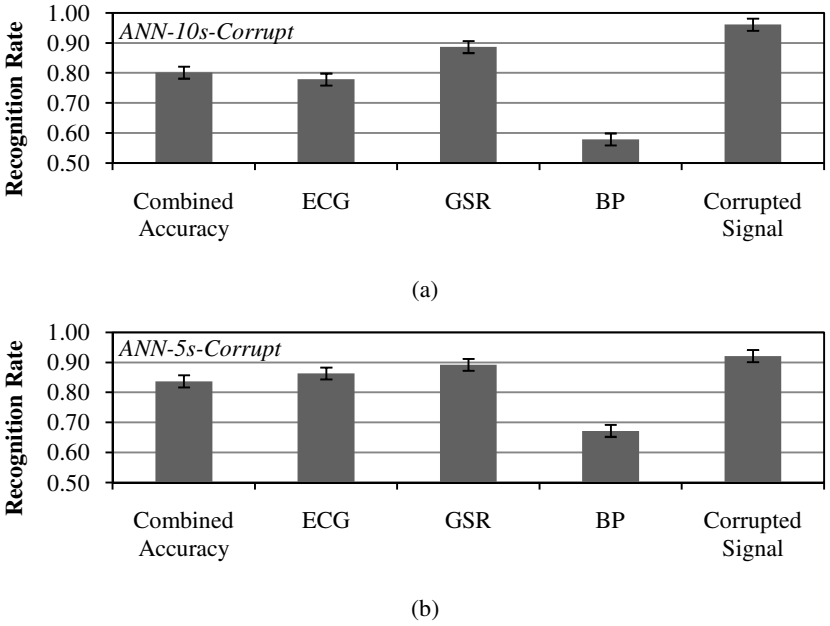
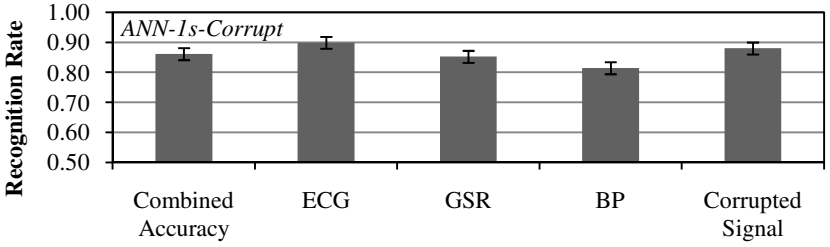
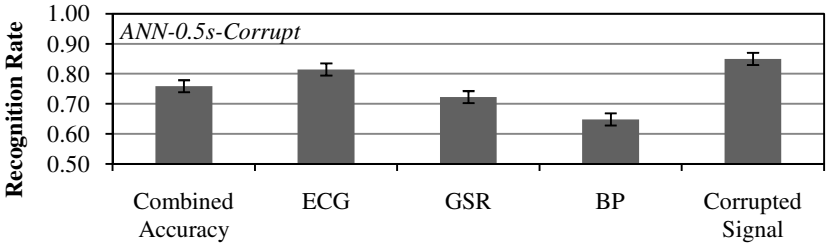


Fig. 4. Recognition rates for the physiological signals and corrupted signals from individual-independent ANN classifiers based on 10-fold cross-validation (a) ANN-10s-Corrupt (b) ANN-5s-Corrupt (c) ANN-1s-Corrupt (d) ANN-0.5s-Corrupt



(c)



(d)

Fig. 4. (*Continued*)

The results in Fig. 2 and Fig. 4 show that the best signal classification rates are achieved when signals segmented into one second time lengths are modeled. Signals that spanned 0.5 seconds did not have sufficient data in input tuples for ANNs to recognize patterns that distinguished one type of signal from the others as well as signals that spanned one second. ANNs that modeled signals which spanned more than one second learnt less general and poorer relationships between data in the signals for the time lengths and the signal class type.

Further, the different topologies of the hidden layers of the ANNs did not show a statistical difference between the classification results according to the Student's T-test ($p > 0.1$). Future work could investigate optimizing the topology of the ANNs including investigating recurrent ANNs and time-delay ANNs for signal classification.

5 Conclusion and Future Work

Different physiological signals were modeled and classified by individual-independent ANNs and ANNs for a particular individual. Signals spanning various time lengths were modeled. Results showed that the highest accuracy values for physiological signals without corrupted signal recognition were produced by the ANNs that modeled signals with a span of one second. However, corrupted signal recognition rates were the highest for ANNs that modeled signals spanning 10 seconds.

Future work can investigate developing an ANN that classifies signals using signals of different time lengths and produce classification rates that are highest for both physiological signals and corrupted signals. Alternatively, it may be beneficial to develop a system that uses a sampling frequency recognizing corruption which is different to the best frequency for recognizing physiological signals. The latter would be better for cutting through the noise which is in some ways the converse of recognizing corrupted signals. Further, the proposed classification system can be extended to model and recognize other types of physiological signals and applied to automatic online signal classification.

References

[1] Jacobson, M.: Analysis and classification of physiological signals using wavelet transforms. In: International Conf. on Electronics, Circuits and Systems, pp. 906–909 (2003)
[2] Killgore, W.D., et al.: Insomnia-related complaints correlate with functional connectivity between sensory–motor regions. NeuroReport 24, 233–240 (2013)
[3] Sharma, N., Gedeon, T.: Objective measures, sensors and computational techniques for stress recognition and classification: A survey. Computer Methods and Programs in Biomedicine 108, 1287–1301 (2012)
[4] Wagner, J., Kim, J., André, E.: From physiological signals to emotions: Implementing and comparing selected methods for feature extraction and classification. In: Int. Conf. on Multimedia and Expo, pp. 940–943 (2005)
[5] Gouizi, K., Bereksi Reguig, F., Maaoui, C.: Emotion recognition from physiological signals. Journal of Medical Engineering & Technology 35, 300–307 (2011)
[6] Sweeney, K.T., et al.: Intelligent artifact classification for ambulatory physiological signals. In: Int. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 6349–6352 (2010)
[7] Sharma, N., Gedeon, T.: Modeling Stress Recognition in Typical Virtual Environments. In: Int. Conf. on Pervasive Computing Technologies for Healthcare (2013)
[8] Healey, J.A., Picard, R.W.: Detecting stress during real-world driving tasks using physiological sensors. IEEE Transactions on Intelligent Transportation Systems 6, 156–166 (2005)
[9] Bersak, D., et al.: Intelligent biofeedback using an immersive competitive environment. In: Designing Ubiquitous Computing Games Workshop, USA (2001)
[10] Shi, Y., et al.: Galvanic skin response (GSR) as an index of cognitive load. In: CHI 2007 Extended Abstracts on Human Factors in Computing Systems, USA, pp. 2651–2656 (2007)
[11] Lin, T., et al.: Do physiological data relate to traditional usability indexes? In: Australia Conference on Computer-Human Interaction, pp. 1–10 (2005)
[12] Miller, L.H., Shmavonian, B.M.: Replicability of two GSR indices as a function of stress and cognitive activity. Journal of Personality and Social Psychology, 753–756 (1965)
[13] McEwen, B.S., Sapolsky, R.M.: Stress and cognitive function. Journal of Current Opinion in Neurobiology 5, 205–216 (1995)

Investigation of Chronic Stress Differences between Groups Exposed to Three Stressors and Normal Controls by Analyzing EEG Recordings

Na Li[1], Bin Hu[1,2,*] , Jing Chen[1], Hong Peng[1], Qinglin Zhao[1], and Mingqi Zhao[1]

[1] The School of Information Science and Engineering,
Lanzhou University, Lanzhou, 730000, China
[2] The School of Computing, Telecommunications and Networks, Birmingham City University,
Birmingham, B42 2SU, UK
{lina2011,bh,jchen10,pengh,qlzhao,zhaomq11}@lzu.edu.cn

Abstract. Despite clear evidence of connections between chronic stress, brain patterns, age and gender, few studies have explored stressor differences in stress detection. This paper presents a stressor-specific evaluation model conducted between stress levels and electroencephalogram(EEG) features. The overall complexity, chaos of EEG signals, and spectrum power of certain EEG bands from pre-frontal lobe(Fp1, Fp2 and Fpz) was analyzed. The results showed that different stressors can lead to varying degree of changes of frontal EEG complexity. Future study will build the stressor-specific evaluation model under considering the effects of gender and age.

Keywords: Stress, Stressor, Electroencephalogram, Complexity, Frontal Asymmetry.

1 Introduction

All of us are exposed frequently to a stressful situation at the societal, community and interpersonal levels. Recently, the number of stress victims is growing at an alarming rate with millions of people on stress relief medication [1].Chronic stress is considered to be the most harmful in people's daily lives and affects people more intensely than other types of stress. Research has consistently demonstrated that chronic stress increases risk for developing a number of negative mental health outcomes such as depression [2, 3].

Most important is how the persons cope up with stress and the brain responds to it. Hence utilizing effective tools to detect stress at an early date is of great significance. Though psychology instruments and hormone for stress identification are utility, the disadvantages of conventional methods are apparent either. As an objective and non-invasive brain function measurement, electroencephalogram (EEG) is an useful tool for investigation of physiological and psychological functions. And quantified

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 512–521, 2013.

EEG (qEEG) analysis methods have played an important role in clinical diagnosis and brain research [4].

EEG alpha asymmetry has been used as a measure of cortical activity linked to emotional process, mood, and psychopathology for years. The frontal cortex is particularly critical in emotional processing [5]. The left hemisphere is more involved in the processing of positive emotions and active behavior, whereas the right hemisphere is more involved in the processing of negative emotions and withdrawal behavior[5]. Consequently, it has been hypothesized that relatively greater right anterior EEG activity may predict the development of psychopathology, specifically anxiety and depression. In [6], a closely link between resting frontal EEG asymmetry and depression and anxiety was demonstrated. Researchers investigated the stability of resting frontal EEG asymmetry in depression in [7], suggesting that resting EEG alpha asymmetry can be reliably assessed in clinical depressed individuals.

During the past years, nonlinear dynamics, such as the correlation dimension (D2), the first positive Lyapunov exponent (L1) and LZ-complexity (LZC) of EEG, have been found valid in detecting changes under different physiological or psychological states. Most of the previous work on analyzing EEG from depressed patients is based on linear methods. Nonlinear analyses of EEG from depressed patients are rarely employed. In [8] researchers demonstrated a link between a decrease of D2 and the symptom of depression, implying that nonlinear tools could be utility in the study of chronic stress.

As a ubiquitous application, Online Predictive Tools for Intervention in Mental Illness (OPTIMI) has been developing tools for stress level prediction through early identification of the onset by monitoring poor stress behavior [9, 10]. As one of the partners of OPTIMI we focus on researching chronic stress identification using EEG. The purpose of this study is stress detection from three groups faced with different social stressors through analyzing EEG recordings. EEG features (extracted from raw EEGs) including linear features such as absolute power, relative power, max power, mean frequency and asymmetry indices as well as nonlinear dynamical measures, namely C0-complexity (C0), LZ-complexity (LZC), correlation dimension (D2), Renyi Entropy (RE) and the first positive Lyapunov exponent (L1) were employed in our study. we are trying to answer the following three questions: (1) which EEG feature can differentiate the stress group from normal ones efficiently; (2) differences of EEG pattern among three stress groups faced with different stressors; (3) hemispheric differences specifically in power values of theta, alpha and beta bands in the stress group.

2 Methods

2.1 Participants

53 right-handed participants volunteered to take part in the study. 18 unemployed men aged from 21 to 41 (mean age = 32.9; S.D. = 7.9) were recruited in group 1.

And participants of group 2 were students aged from 20 to 35 (mean age = 21.9; S.D. = 6.1) at risk of chronic stress due to frequent examinations and graduation pressure, while group3 were mothers of retarded children aged 30-52 (mean age = 36.4; S.D. = 7.6). All the participants were free of prior history of psychopathology, cardiovascular or medications with a potentially negative impact on the heart and using of medication affecting mood. Participants filled in a scale of Beck Depression Inventory (BDI) first. Individuals with a BDI score below 10 were comprised in the control group, while individuals with a BDI score of 10 or higher were included in the stress group [11]. The number of volunteers in the stress and control group was shown in Table1.

Table 1. The number of volunteers in the stress and control group

	Stress group	**Control group**
Group 1	9	9
Group 2	7	8
Group 3	9	11

2.2 EEG Recording

Participants were seated in a comfortable sound and light attenuated room. Data collection, which lasted two minutes for each person, was made while participants kept relaxing state with their eyes closed. Resting EEG was collected from three electrodesFp1, Fp2 and Fpz referenced to earlobes according to International 10-20 system. EEG data were sampled at 256Hz. A wearable EEG sensor we developed was utilized in our experiment [12].

2.3 EEG Data Processing

Raw EEG usually contains lots of artifacts due to eye-blink, eyeball movements, facial and body movements, etc. Hence it is necessary to remove these artifacts. Firstly, a low-pass filter with cutoff 40Hz was adopted to eliminate EEG signals drifting and EMG disturbances. Then we eliminated EOG disturbances with the wavelet algorithm.

EEG recordings were segmented into 4-s epochs within 2-s overlap. After removing frequency interference and artifacts, the fast Fourier transform method was utilized to calculate the absolute power (μV^2), relative power (%), max power (μV^2), and mean frequency (Hz) in theta (4-8 Hz), alpha (8-13 Hz) and beta (13-20 Hz) bands on each electrode. Relative power indices for each band were derived by expressing absolute power in each separate band as a percent of the absolute power summed over the four frequency bands. Mean frequency was also derived for the entire spectrum.

A number of studies have reported that greater relative right anterior and posterior EEG activity is related to symptoms of depression and anxiety [6, 13, 14]. Frontal asymmetry is defined as ln(F2)-ln(F1), where higher scores reflect lower left alpha

power, and consequently higher left cortical activation, relative to the right cortex. We computed hemispheric asymmetry of alpha power as well as that of beta and theta bands.

Nonlinear features of EEG contain the complexity of EEG and the chaotic characteristics of the brain. In this paper, we employed methods of nonlinear dynamics, including C0-complexity (C0), LZ-complexity (LZC), the correlation dimension (D2), Renyi spectral (RE)and the largest Lyapunov exponent (L1) on each electrode respectively.

Lempel–Ziv complexity (LZC) tests the randomness of a sequence by searching for patterns in a series. It has been extensively employed in EEG data analysis and proved an effective tool in researching biomedical signals. In [15], researchers proposed an algorithm to generate a given sequence using two fundamental operations, namely: copy and insert by parsing it from left to right. The Lempel-Ziv complexity c(n) of a sequence of length n is given by the shortest sequence generated using the copy and insert operation that can generate the given sequence.

The correlation dimension (D_2) describes the complexity of EEG data in the phase space. We defined the EEG signal as a time sequences X(t), t =1, 2, 3, ...,N, which can be reconstructed into a m-dimensional vector Y_j, j = 1, 2, 3, ...,M(M = N −(m− 1) * time − delay) with time-delay embedding as given in Grassberger and Procaccia (GP) [16]. In order to calculate D2, the correlation integral function should be estimated first [17]:

$$C_m(r) = \frac{2}{(M-1)\times M \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} \left(\theta - (r_{ij})\right)} \quad (1)$$

Where θ is the Heaviside unit function. The Heaviside function is 0 if the distance between the vectors is greater than r, while it is 1 if the distance between the vectors is less than r. Theoretically if m is sufficiently large and r is small enough, we can assume the following relation: $C_m(r) \propto rD$.

The correlation dimension (D2) is then defined as:

$$D_2 = \lim_{r\to 0} \frac{\partial \log(C_m(r))}{\partial \log r} \quad (2)$$

The dimension of the attractor is estimated from the slope of a linear scaling region in the $\log(C_m(r)) - \log r$ plot.

The largest Lyapunov exponent [18] measures the exponential divergence of initially close state-space trajectories and estimates the amount of chaos in a system. If the largest lyapunov exponent L1 <0, standing for two trajectories with nearby initial conditions contract; If L1 >0, standing for two trajectories with nearby initial conditions diverge at an exponential rate and the system sensitivity to initial conditions, also indicating chaos. In this paper, Michael T.Rosenstein et al. [18] algorithm is applied in calculating the largest lyapunov exponent.

The concept of C0-complexity, a description of time sequences randomness was provided in [2]. The dynamic tendency of C0-complextiy to EEG signal agrees with Approximate Entropy. Compared to Approximate Entropy, C0-complexity is mainly

calculating through Fast Fourier Transform (FFT). Extract ing C0-complexity only need running FFT once, so it has advantage of saving a part of the computational effort. Xu et al. [2] put forward C0-complexity at practical analysis of EEG signal processing. The concept of Renyi spectral (generalized entropy) of a probability distribution was introduced by Alfred Rényi [19].

2.4 Statistical Analysis

Statistical analysis was done with SPSS17.0. To compare statistically significant differences in the stress group and control group, an independent-sample t-test was used for each feature at three electrodes. Significance levels were set at $p \leq 0.05$ for all statistical analyses.

3 Results

The results of t-test of each feature are presented in Tables 2-4.

Table 2. The results of t-test for participants of Group 1 Sig value p<0.05 means difference

	Feature	Electrodes		
		Fp1	**Fp2**	**Fpz**
Group1	D2	0.060	**0.009**	**0.039**
	L1	**0.019**	**0.004**	**0.006**
	LZC	**0.002**	**0.001**	**0.000**

Table 3. The results of t-test for participants of Group 2 Sig value p<0.05 means difference

	Feature	Electrodes		
		Fp1	**Fp2**	**Fpz**
Group2	D2	**0.042**	**0.043**	**0.021**
	L1	**0.037**	**0.047**	**0.034**
	LZC	**0.029**	**0.016**	**0.039**

Table 4. The results of t-test for participants of Group 3 Sig value p<0.05 means difference

	Feature	Electrodes		
		Fp1	**Fp2**	**Fpz**
Group3	D_inf	**0.005**	**0.001**	**0.027**
	D_q_0	**0.005**	**0.001**	**0.024**
	D_q_1	**0.006**	**0.002**	**0.021**

For Group 1(unemployed men), the mean difference between groups (stress vs. control) was significant for features L1 and LZC at electrode Fp1. And the difference was significant for D2, L1 and LZC at both Fp2 and Fpz. As to D2 and LZC, the

mean value of the stress group was higher than the control group at all the three electrodes, while the mean value of stress group was lower than control group in Fp2 and Fpz with regard to L1 (Figure 1). For Group 2(students faced with frequent examinations and graduation pressure), there was significantly difference between groups for D2, L1 and LZC at Fp1, Fp2 and Fpz. The mean value of the stress group was higher than the control group for D2 and LZC, while the mean value of stress group was lower than control group for L1(Figure 2).For Group 3(mothers with retarded children), it was obvious that the stressful individuals had higher mean value of D_inf, D_q_0 and D_q_1than normal controls (Figure 3).

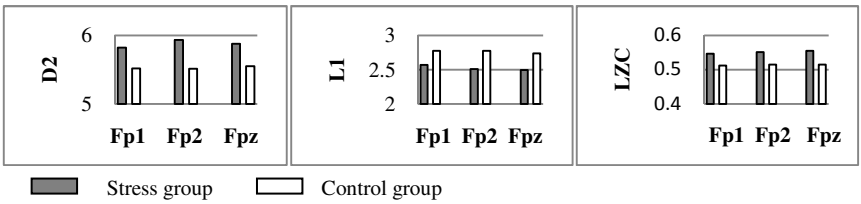


Fig. 1. The mean value of EEG features of Group 1 (the result of t-test is significant)

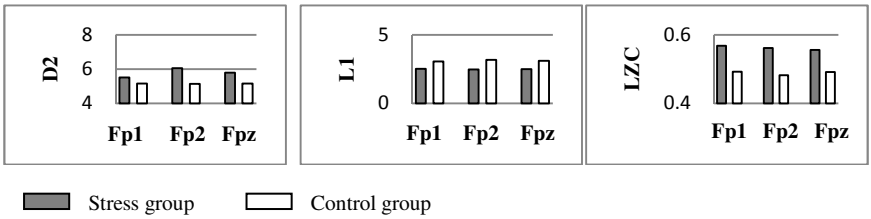


Fig. 2. The mean value of EEG features of Group 2 (the result of t-test is significant)

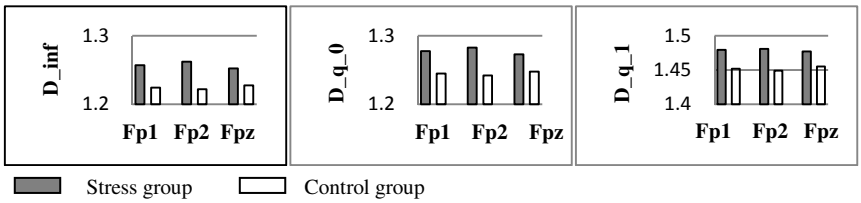


Fig. 3. The mean value of EEG features of Group 3 (the result of t-test is significant)

There was a significant difference between groups (stress vs. control) for hemispheric asymmetry indices at all theta, alpha and beta band ($p<0.01$). The controls had distinctly higher hemispheric asymmetry indices than the stressful subjects (Figure 4).

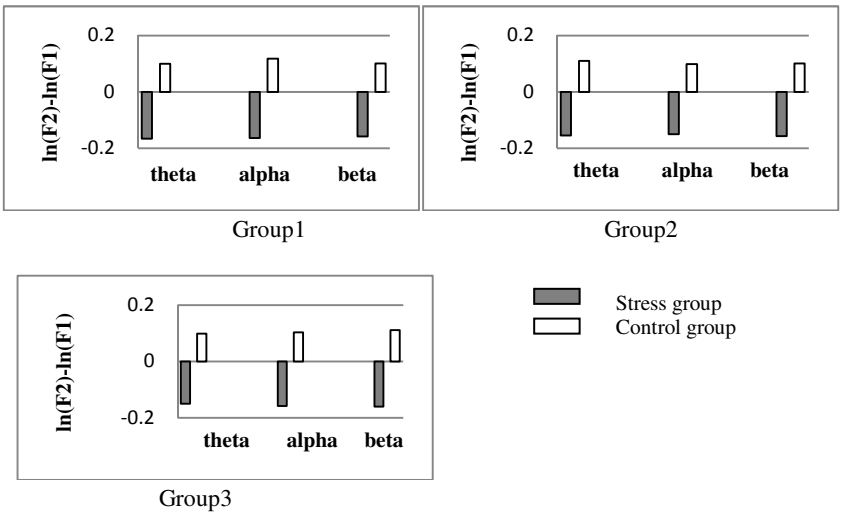


Fig. 4. The mean value of frontal asymmetry indices for theta, alpha and beta band of Group 1, Group2 and Group3

4 Discussion

In this part we discuss the results from two aspects including nonlinear dynamics features and frontal hemispheric asymmetry.

4.1 Nonlinear Dynamics Signal Features

EEG is recognized as a dynamical system, which is neither purely chaotic nor stochastic. Consequently it is reasonable to analyze such signals employing nonlinear features of EEG. For both Group1 and Group2, D2, L1 and LZC can effectively distinguish stressful individuals from normal controls. The stress group presented a significantly higher LZC than the normal control group. A higher LZC implies a greater chance of the occurrence of new sequence patterns and thus a more complex dynamical behavior. We also found a higher D2 in the stress group relative to the control group, opposite to the result provided by Nandrino [8]. The possible reason is that D2 is a measure of dimensional complexity of a chaotic system, while LZC represents the sequence pattern complexity in a dynamic system. Compared to normal controls, the stressful subjects showed significantly lower L1 values, confirming that L1 is effective. As to Group3, mothers of retarded children, the Renyi entropy of the stress group was significantly higher than the control group. Our results were consistent with the finding given by Tang [20], showing the alpha activity of depression patients is more complex during resting.

Similar results were obtained from three stress groups, demonstrating the stress subjects have more complex brain activity than normal controls. Nevertheless, the differences of the results may due to the various stressors subjects bear. Social stress

is typically the most frequent type of stressor that people experience in their daily lives. Stress may arise from one's relationships with others and from the social environment. We recruited subjects undergoing three different stressors, including unemployed men, students faced with frequent examinations and graduation pressure and mothers of retarded children. The results of Group 1 and Group 2 were similar except Group 3, which may be caused by various experiences they underwent. Different stressors can lead to varying degrees of emotional, behavioral and physiological changes as well as brain function, reflecting in complexity of frontal EEG. The long-term effect of chronic stress may produce diverse changes in brain function. This can explain the distinction among groups. Future study will investigate this issue in depth.

4.2 Frontal Hemispheric Asymmetry

As said before numbers of studies have reported that greater relative right anterior and posterior EEG activity is related to symptoms of depression and anxiety. Recent fMRI studies have also reported that depression shows different patterns of hemispheric activations in the superior frontal areas [21]. Frontal asymmetry indices of EEG power have been put forward as a biological indicator for depression for years. In our study, frontal hemispheric power asymmetry indices of alpha band were employed as measures as well as theta and beta power asymmetry indices. There was a significant difference on power asymmetry of three bands between stressful individuals and normal controls in all the three groups faced with different kinds of stressors. Stressful subjects had negative hemispheric asymmetry indices, and the controls were the opposite, implying greater relative right anterior EEG activity in the stressful subjects. The finding in our study is in keeping with some results in other researches like [13, 14].

Much of the previous literatures came to the finding that frontal asymmetry was related to depression restricted to females [14]. This can be observed in several studies [22-25]. Gender difference was an issue in this research. Some literatures exposed that increased right frontal activity was only found in depressed females and no effects in males [26]. Coan [27] proposed that gender difference might exist in frontal asymmetry and negative emotionality owing to some genes issues.

Additionally, previous results showed frontal asymmetry closely related to age in individuals at the risk of depression. Alessandro Carvalho et al. [11] found no difference regarding the alpha frontal asymmetry in depressed, remitted and non-depressed elderly subjects. Findings from functional neuroimaging studies in elderly people tend to find in most cases a weaker and more diffuse cortical activation, with reduced hemispheric asymmetry [28]. Similar results were also found in some other studies, observing no differences in frontal asymmetry between depressive people and normal controls [29, 30]. A possible explanation is a structural and functional impairment of the right hemisphere with the aging process [11]. Our future research will carefully consider the factors of both gender and age into the present study.

5 Conclusion

Our findings demonstrate EEG nonlinear dynamics features are effective measures to detect chronic stress. EEG frontal asymmetry of theta, alpha and beta bands can be

biological indicators for chronic stress, showing relative greater right anterior EEG activity in stressful individuals, which is consistent with previous researches. In addition, different stressors can lead to varying degrees of emotional, behavioral and physiological changes, reflecting in complexity of frontal EEG. Consequently, analysis of chronic stress according to the specific stressor is requisite.

Acknowledgements. This work was supported by the National Basic Research Program of China (973 Program, No.2011CB711000), the National Natural Science Foundation of China under Grant 60973138 and Grant 61210010, the EU's Seventh Framework Programme OPTIMI under Grant 248544, Program of International S&T Cooperation of MOST under Grant 2013DFA11140, the Central Universities Fundamental Research Funds under Grant lzujbky-2012-39 and lzujbky-2013-45, and Science and technology plan projects in Gansu province 1208RJZA127.

References

1. Hoffmann, E.: Brain Training Against Stress: Theory, Methods and Results from an Outcome Study. Stress Report 4, 1–24 (2005)
2. Chen, F., et al.: A New Measurement of Complexity for Studying EEG Mutual Information. Shengwu Wuli Xuebao 14(3), 508–512 (1998)
3. Mnnroe, S.M., Slavich, G.M., Georgiades, K.: The Social Environment and Life Stress in Depression. In: Handbook of Depression, pp. 340–360 (2009)
4. Thakor, N.V., Tong, S.: Advances in Quantitative Electroencephalogram Analysis Methods. Annual Review of Biomedical Engineering 6(1), 453–495 (2004)
5. Davidson, R.J., et al.: Depression: Perspectives from Affective Neuroscience. Annual Review of Psychology 53(1), 545–574 (2002)
6. Thibodeau, R., Jorgensen, R.S., Kim, S.: Depression, Anxiety, and Resting Frontal EEG Asymmetry: A Meta-analytic Review (0021-843X (Print))
7. Allen, J.J.B., et al.: The Stability of Resting Frontal Electroencephalographic Asymmetry in Depression. Psychophysiology 41(2), 269–280 (2004)
8. Nandrino, J.-L., et al.: Decrease of Complexity in EEG as a Symptom of Depression. NeuroReport 5(4), 528–530 (1994)
9. Peng, H., et al.: A Method of Identifying Chronic Stress by EEG. Personal and Ubiquitous Computing, 1–7 (2012)
10. Zhang, X., Hu, B., Moore, P., Chen, J., Zhou, L.: Emotiono: An Ontology with Rule-Based Reasoning for Emotion Recognition. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part II. LNCS, vol. 7063, pp. 89–98. Springer, Heidelberg (2011)
11. Carvalho, A., Moraes, H., et al.: EEG Frontal Asymmetry in the Depressed and Remitted Elderly: Is It Related to the Trait or to the State of Depression? (1573-2517 (Electronic))
12. Bin, H., et al.: EEG-Based Cognitive Interfaces for Ubiquitous Applications: Developments and Challenges. IEEE Intelligent Systems 26(5), 46–53 (2011)
13. Schaffer, C.E., Davidson, R.J., Saron, C.: Frontal and Parietal Electroencephalogram Asymmetry in Depressed and Nondepressed Subjects. Biological Psychiatry 18, 753–762 (1983)
14. Smit, D.J.A., et al.: The Relation between Frontal EEG Asymmetry and the Risk for Anxiety and Depression. Biological Psychology 74(1), 26–33 (2007)

15. Lempel, A., Ziv, J.: On the Complexity of Finite Sequences. IEEE Transactions on Information Theory 22(1), 75–81 (1976)
16. Grassberger, P., Procaccia, I.: Measuring the Strangeness of Strange Attractors. Physica D: Nonlinear Phenomena 9(1-2), 189–208 (1983)
17. Stam, C.J., van Woerkom, T.C.A.M., Pritchard, W.S.: Use of Non-linear EEG Measures to Characterize EEG Changes during Mental Activity. Electroencephalography and Clinical Neurophysiology 99(3), 214–224 (1996)
18. Rosenstein, M.T., Collins, J.J., De Luca, C.J.: A Practical Method for Calculating Largest Lyapunov Exponents from Small Data Sets. Physica D: Nonlinear Phenomena 65(1-2), 117–134 (1993)
19. Kulish, V., Sourin, A., Sourina, O.: Human Electroencephalograms Seen As Fractal Time Series: Mathematical Analysis and Visualization. Computers in Biology and Medicine 36(3), 291–302 (2006)
20. Tang, Y., et al.: Entropy Analysis of the EEG Alpha Activity in Depression Patients. Journal of Biomedical Engineering 26(4), 739–742 (2009)
21. Hugdahl, K.R., Rishovd, B., Lund, A., Asbjørnsen, A., Egeland, J., Ersland, L., Landrø, N.I., Roness, A., Stordal, K.I., Sundet, K., Thomsen, T.: Brain Activation Measured With fMRI During a Mental Arithmetic Task in Schizophrenia and Major Depression. The American Journal of Psychiatry 161(2), 286–293 (2004)
22. Field, T., Pickens, J., et al.: Targeting Adolescent Mothers with Depressive Symptoms for Early Intervention (0001-8449 (Print))
23. Debener, S., et al.: Is Resting Anterior EEG Alpha Asymmetry a Trait Marker for Depression? Findings for Healthy Adults and Clinically Depressed Patients. Neuropsychobiology 41(1), 31–37 (2000)
24. Bruder, G.E., et al.: Electroencephalographic and Perceptual Asymmetry Differences between Responders and Nonresponders to an SSRI Antidepressant. Biological Psychiatry 49(5), 416–425 (2001)
25. Hinrikus, H., Suhhova, A., et al.: Spectral Features of EEG in Depression (1862-278X (Electronic))
26. Tomarken, A.J., et al.: Resting Frontal Brain Activity: Linkages to Maternal Depression and Socio-economic Status among Adolescents. Biological Psychology 67(1-2), 77–102 (2004)
27. Coan, J.A., et al.: The Heritability of Trait Frontal EEG Asymmetry and Negative Emotionality: Sex Differences and Genetic Nonadditivity. The University of Arizona (2003)
28. Minati, L., Grisoli, M., Bruzzone, M.G.: MR Spectroscopy, Functional MRI, and Diffusion-tensor Imaging in the Aging Brain: A Conceptual Review. Journal of Geriatric Psychiatry and Neurology 20(1), 3–21 (2007)
29. Bruder, G.E., et al.: Regional Brain Asymmetries in Major Depression with or without an Anxiety Disorder: A Quantitative Electroencephalographic Study. Biological Psychiatry 41(9), 939–948 (1997)
30. Reid, S.A., Duke, L.M., Allen, J.J.B.: Resting Frontal Electroencephalographic Asymmetry in Depression: Inconsistencies Suggest the Need to Identify Mediating Factors. Psychophysiology 35(4), 389–404 (1998)

A Novel Hybrid SCH-ABC Approach for the Frequency Assignment Problem

Gang Yang*,**, Shaohui Wu, Jieping Xu, and Xirong Li

Multimedia Computing Lab, School of Information,
Renmin University of China, China
{yanggang,shaohuiwu,xjieping,xirong}@ruc.edu.cn

Abstract. This paper proposes an efficient hybrid approach based on the stochastic competitive Hopfield neural network(SCHNN) and artificial bee colony (ABC), which named SCH-ABC. The hybrid algorithm aims to cope with the frequency assignment problem (FAP). The objective of FAP is to minimize the cochannel interference between satellite communication systems by rearranging the frequency assignments so that they can accommodate the increasing demands. In fact, as our SCH-ABC algorithm owns good adaptability, it can not only deal with the frequency assignment problem, but also cope with other problems including the clustering, classification, the maximum clique problem etc. With the help of hybridization, SCH-ABC makes up for the defects in the Hopfield neural network and ABC while fully utilizing the advantages of the two algorithms.

Keywords: Frequency assignment problem, artificial bee colony, neural network, hybrid algorithm.

1 Introduction

The frequency assignment problem (FAP) is a famous problem due to its widely applications including satellite communication systems, mobile telephone and TV broadcasting. In satellite communication systems, the reduction of the cochannel interference has become a major factor for determining system design [1], [2]. Furthermore, due to the necessity of accommodating as many satellites as possible in geostationary orbit, this interference reduction has become an even more important issue with the increasing number of geostationary satellites [3]. To deal with interference reduction in practical situations, the rearrangement of frequency assignments is considered as an effective measure [4].

FAP is a NP-complete combinatorial optimization problem and many approaches have been proposed to tackle it [5]. The application of neural networks in frequency assignment problems was first proposed Kunz [6]. Then

* Corresponding author.

** This research was partially supported by the grants from the Natural Science Foundation of China (No.61003205); the Qianjiang Talent Project of Zhejiang Province (No.2011R10087).

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 522–529, 2013.

Funabiki and Nishikawa(1997) proposed a gradual neural network(GNN). After that, Salcedo-Sanz et al. (2004) combined a binary Hopfield neural network with simulated annealing(HopSA) for the FAP[2]. The disadvantages of these algorithms are their heavy computation burden and excessive computation time. Wang et al. (2011) proposed a stochastic competitive Hopfield neural network(SCHNN)[7]. They introduced stochastic dynamics to help the network escape from local minima, but the way only using the dynamics is not efficient. Recently, modeling the behavior of social insects including bees, birds and ants has become an emerging area for the optimization problems[8]. Among these algorithms, the artificial bee colony(ABC) algorithm is the promising one. It has obtained better performances on many problems than other well-known modern heuristic algorithms such as genetic algorithm, differential evolutional algorithm and particle swarm optimization algorithm[9]. However, it also has many disadvantages including prematurity and the low searching speed.

The contributions of this paper are: (1) a novel hybrid SCH-ABC approach is proposed that can make up for the defects in the neural network and ABC while improving their abilities to search better solutions; (2) the proposed SCH-ABC obtains better performance than other algorithms compared in this paper, which is the contribution to the available literature on the FAP.

The rest of the paper is organized as follows: in the next section we define and analyze the FAP. In section 3, our hybrid algorithm is described in detail. Experiments and results are shown in Section 4. Finally, Section 5 ends the paper with concluding remarks.

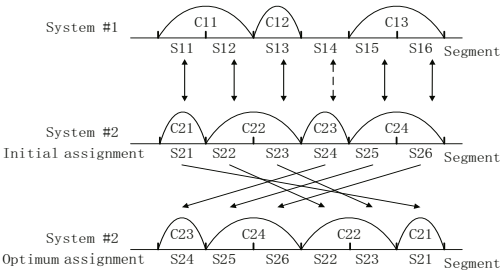
2 Problem Definition

In this section, the FAP in satellite communications systems can be described as a combinatorial optimization problem with three constraints and two objectives [10]. Given two adjacent satellite systems, FAP consists of reducing the inter-system cochannel interference by rearranging the frequency assignment on carriers in system #2(M segments, N carriers), while the assignment in system #1(M segments) remains fixed(Fig.1(a)). The interference between two M-segment systems is described by a $M \times M$ interference matrix IM(Fig.1(b)), in which the ijth element e_{ij} stands for the cochannel interference when segment i in system #2 uses a common frequency with segment j in system #1.

There are three constraints in FAP(Funabiki & Nishikawa, 1997)[10]:

(1)Every segment in system #2 must be assigned to a segment in system #1. (2)Each segment in system #1 can be assigned by at most one segment in system #2. (3)All segments of each carrier in system #2 must be assigned to consecutive segments in system #1 in the same order.

The two objectives are shown as follows: (1)Minimize the largest element of the interference matrix selected in the assignment. (2)Minimize the total interference of all the selected elements. Note that the first objective has a higher priority over the second objective.



(a) Cochannel interference model.

		C11		C12		C13	
		S11	S12	S13	S14	S15	S16
C21	S21	20	20	40	0	25	25
C22	S22	50	10	30	0	55	∞
	S23	∞	50	30	0	15	55
C23	S24	30	30	45	0	35	35
C24	S25	45	5	25	0	50	∞
	S26	∞	45	25	0	10	50

(b) Interference matrix *IM*.

Fig. 1. Graphs of the cochannel interference model and interference matrix

3 Our hybrid SCH-ABC for FAP

Based on the stochastic competitive Hopfield neural network(SCHNN) and artificial colony bee(ABC), three hybridizations between the two algorithms are explored. Linear structure has been taken in order to make the hybridization owning favorable commonality, which can be helpful for hybridizations between Hopfield-type neural networks and other evolutionary algorithms.

The first hybridization aims to improve the ability of bee colony to search better food sources by adding solutions of neural network as high quality scouts. As the scouts in bee colony just search food source randomly, the solutions from neural network may help the bee colony find better food sources. The second hybridization is to help the neural network to escape from local minima efficiently by bee colony. The solutions in neural network will be compared with the best food source in colony. If the solution is worse, it will be replaced by the best food source. The third hybridization combines the first two hybridizations. Not only the solutions in neural network will be added to colony as high quality scouts, but also the best food sources in colony may help neural network to escape from local minima efficiently. In our research, the third hybridization obtains better performance than the other two hybridizations. Then we select the third hybridization for SCH-ABC algorithm and show it in Algorithm 1.

In Algorithm 1, the parameters will be initialized first. Then the neural network named SCH in our hybrid algorithm updates to search solutions for FAP. In SCH, a total energy function proposed for the first two constraints is shown as follows.

$$E = \frac{A}{2}\sum_{i=1}^{N}(\sum_{q=1}^{M}V_{iq}-1)^2 + \frac{B}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{\substack{p=1\\p\neq i}}^{N}\sum_{q=j-c_p+1}^{j+c_i-1}V_{ij}V_{pq} \tag{1}$$

where A and B are coefficients. As the neural network itself satisfies the third constraint, the neural network has met all the three constraints when E becomes zero [7].

Algorithm 1. SCH-ABC Algorithm

```
Initialize the parameters of SCH-ABC;
Set t=0;
while t < the max number of iterations or stability criteria are not satisfied do
  for i = 1 to N do
    for j = 1 to M do
      s=⌊t/N⌋;
      Compute the inputs of neurons with Eq.2;
    end for
    for j = 1 to M do
      Update the outputs of neurons with Eq.4;
    end for
  end for
  Move the employed bees onto the food sources;
  Move the onlookers onto the food sources;
  if A food source is not improved by limit times then
    the food is abandoned by its employed bee;
    the employed bee is converted to a scout;
  end if
  Add the solution of neural network into the bee colony as a scout;
  Move the scouts onto new food sources randomly;
  Memorize the best food source found so far;
  if the solution of neural network is worse than the best food source then
    the outputs will be replaced by the food source;
  end if
  t = t+1;
end while
```

SCH satisfies the rule that large carriers with many segments should be assigned as early as possible, or else, it would be difficult to assign them after many carriers have been already assigned [10]. Thus, the inputs of the neurons are modified by

$$u_{ij}(t+1) = -W_2 \sum_{\substack{p=1 \\ p \neq i}}^{N} \sum_{q=\max(j-c_p+1,M)}^{\min(j+c_i-1,M)} c_p v_{pq}(t) - W_3 d' \tag{2}$$

where W_2 and W_3 are weighting factors.

In SCH, the outputs of the neurons are modified by

$$u'_{ij}(t+1) = \alpha(s) \cdot u_{ij}(t+1) \tag{3}$$

$$v_{ij}(t+1) = \begin{cases} 1, u'_{ij} = \max\limits_{k=1,\ldots M} \{u'_{ik}(t+1)\} \\ 0, otherwise \end{cases} \tag{4}$$

where s is the updating step number, α(s) is a random multiplier, and $u'_{ij}(t+1)$ is the transient variable. The multiplier $\alpha(s)$ in Eq.3 is given by

$$\alpha(s) = random(h(s), 1) \tag{5}$$

where $h(s) = 1 - Q \cdot e^{-s/\lambda}$. Q just is a parameter.

SCH obtains good performance for FAP, but it still has many disadvantages. Its ability for escaping from local minima only utilizing the stochastic dynamics is not strong. At the same time, the two objectives of FAP are not fully synchronized which means that a result with a smaller value of the largest interference may have a larger total interference. There is no good solution in SCH for this phenomenon. In SCH-ABC, we select the artificial bee colony(ABC) algorithm to help SCH to deal with these disadvantages of SCH.

ABC is one of the most recently introduced swarm-based algorithms and it has been used for optimizing many problems. In ABC, food sources standing for the solutions of FAP are found by bees. The three types of bees in ABC are the employed bees, onlookers and scouts. The employed bees are associated with a food source which they are currently exploiting. The scouts search the environment surrounding the nest for new food sources and onlookers wait in the nest and establish a food source through the information shared by employed foragers [9]. In our SCH-ABC algorithm, when the updating of neural network ends, bees in colony start to search solutions for FAP.

The main steps for searching solutions by bees are given as follows: N_e, N_o, N_s standing for the number of employed bees, onlookers and scouts respectively are initialized and the parameter *limit* is determined in the first step of SCH-ABC. After the updating of neural network, the employed bees are randomly moved onto different food sources which have met the three constrains of FAP. Then the nectar amounts of food sources associated with the employed bees are calculated. Specifically, the nectar amounts are the reciprocals of the largest interferences. If two food sources have the same amounts, their reciprocals of the total interferences will be compared. Thus, the smaller interference means more nectar amounts. The probability values for food sources with which they are preferred by the onlooker bees will be calculated. In step 14 in Algorithm 1, the onlooker bees are moved onto new food sources and their nectar amounts will be calculated. If a food source is not improved by *limit* times, it will be abandoned by its employed bee and the employed bee is converted to a scout. Afterwards, the solution of neural network will be added into bee colony as scouts and the scouts in colony will be sent into the search area for discovering new food sources, randomly. Then the best food source discovered so far will be memorized.

When searching food sources by bees is over, SCH-ABC will check whether the neural network gets trapped in local minima. As shown in step 22 in Algorithm 1, if the solution of neural network is worse than best food source in bee colony, it will be replaced by that food source. By the bee colony, the neural network can escape from local minima efficiently. The cycles increase until the requirements are met or it runs to the max number of iterations. Finally, the best food source in bee colony is the best solution of SCH-ABC algorithm.

In SCH-ABC, SCH improves the ability of bee colony to search better solutions and the bee colony also help SCH to escape from local minima efficiently.

By combining the SCH and ABC, SCH-ABC remedies the shortcomings of the two algorithms while fully utilizing their advantages.

4 Simulation Results and Discussions

In order to test the performance of SCH-ABC, simulations were implemented in Matlab on a PC(Core(TM) i5-3450 3.10GHz, 8.0G RAM). In our experiments, SCH-ABC was compared with other algorithms on 5 benchmark problems and 12 large problems randomly generated. Our SCH-ABC algorithm obtains better or comparable performance on all cases.

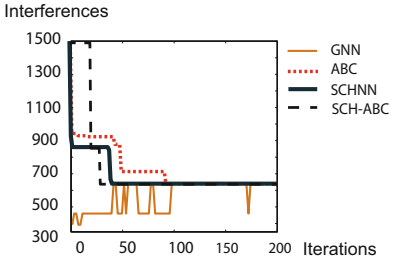
Table 1 shows the detailed results obtained by HopSA, GNN, SCHNN, ABC and SCH-ABC on 5 benchmark problems which were called instances 1-5 respectively in Funabiki and Nishikawa(1997) [10]. In Table 1, each result includes the largest interference and total interference. Since BM1-BM5 are benchmark problems, the results of GNN on BM1-BM5 are directly from Funabiki and Nishikawa(1997)[10], the results of the HopSA are directly from Salcedo-Sanz et al.(2004)[2] and the results of SCHNN are directly from Wang et al(2011)[7]. Note that only the best values are shown in Table 1.

Table 1. Comparison of different algorithms for benchmark problems BM1-BM5. Our SCH-ABC approach obtains better or comparable results than other algorithms.

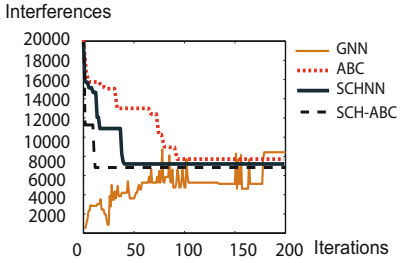
Group	Instance	HopSA		GNN		SCHNN		ABC		SCH-ABC	
		Largest	Total	Largest	Total	Largest	Total	Largest	Total	Largest	Total
Group 1	BM1	30	100	30	100	30	100	30	100	30	100
	BM2	4	13	4	13	4	13	4	13	4	13
	BM3	7	85	7	85	7	85	7	85	7	85
	BM4	64	880	64	880	64	880	64	873	64	**855**
	BM5	817	6910	640	8693	640	7243	640	7335	640	**7006**

As can be seen from Table 1, the five algorithms obtain the same results on BM1-BM3. On BM4, the five algorithms obtain the same values of the largest interference and the ABC and SCH-ABC obtain smaller values for the total interference. On BM5, the total interference of HopSA is the smallest, but its value of the largest interference element is larger than other algorithms. As the first objective of decreasing the largest interference has a higher priority, the performance of HopSA is not good. The performance of SCH-ABC is the best among these algorithms. It shows that the hybridization between SCHNN and ABC can help the neural network escape from local minima and help the bee colony find better results.

As the performance of HopSA is worse than other algorithms, only the evolutions of GNN, SCHNN, ABC and SCH-ABC on BM5 are shown in Fig.2. In the evolution process, two characteristics can be found in Fig.2. The first characteristic is that the largest and total interferences for GNN increase continually.



(a) Evolutions for the largest interference on BM5.



(b) Evolutions for the total interference on BM5.

Fig. 2. Evolutions of the four algorithms for the largest and total interferences on BM5

Table 2. Comparison of the results obtained by SCHNN, ABC, GNN and SCH-ABC for 12 large problems randomly generated

Group	Instance	SCHNN		ABC		GNN		SCH-ABC	
		Largest	Total	Largest	Total	Largest	Total	Largest	Total
Group 2	Case 6	9	882	9	897	9	933	8	778
	Case 7	92	7481	94	8362	95	8531	85	6532
	Case 8	939	78910	963	83269	982	89326	913	77052
Group 3	Case 9	92	5643	96	6209	99	7149	87	5136
	Case 10	96	11076	98	13556	99	15232	94	10017
	Case 11	97	16839	98	19931	99	21367	95	17694
	Case 12	98	18542	99	20149	99	23524	97	18077
	Case 13	99	22685	99	24065	99	25986	98	22336
	Case 14	99	26398	99	28627	99	29473	98	21531
Group 4	Case 15	92	20812	95	25343	97	26394	**85**	19968
	Case 16	91	11551	94	17874	97	20305	**82**	**10545**
	Case 17	84	3381	89	4237	93	5749	**73**	3159

In the early stage evolution, the outputs of GNN do not satisfy the three constraints and the majority of outputs are zeros. Thus, both the largest and the total interferences are very small. Then the outputs satisfy the three constraints gradually and the interferences increase with the evolution of the neural network. The second characteristic is that both the largest interference and total interference of SCHNN, ABC and SCH-ABC become smaller and smaller. The SCH-ABC is more likely to find better solutions with less iterations. In SCH-ABC, when the neural network gets trapped in local minima, the food source can help the neural network to escape from it efficiently. And the neural network as a scout also can improve ability of the bee colony to search better food sources.

We also compare SCH-ABC with SCHNN, ABC and GNN on 12 large problems(Case 6-17) randomly generated. The detailed results can be found in Table 2. From Table 2, we observe that the advantages of SCH-ABC are more obvious on large problems. As the hybridization can help the neural network and bee colony to find better solutions with less iterations, SCH-ABC can cope with large problems very well.

5 Conclusion

In this paper, we propose a novel hybrid approach SCH-ABC based on the stochastic competitive Hopfield neural network(SCHNN) and artificial bee colony(ABC) to tackle the frequency assignment problem(FAP). With the help of hybridization, SCH-ABC can help the neural network to escape from local minima efficiently and help the bee colony to find better solutions. As SCH-ABC is very robust, it can not only deal with the frequency assignment problem, but also tackle with other problems including the clustering, classification, the maximum clique problem etc. In the experiments, we test our SCH-ABC on 5 benchmark problems and 12 large problems randomly generated. SCH-ABC obtains better or comparable performance on all cases. Since the SCH-ABC algorithm is a general optimization algorithm, our future work will include applications of SCH-ABC to other practical optimization problems.

References

1. Wang, L., Member, S.: Noisy Chaotic Neural Network With Variable Thresholds for the Frequency Assignment Problem in Satellite Communications. IEEE Transactions on Systems, Man, and Cybernetics 38(2), 209–217 (2008)
2. Salcedo-Sanz, S., Santiago-Mozos, R., Bousoño Calzón, C.: A hybrid hopfield network-simulated annealing approach for frequency assignment in satellite communications systems. IEEE Transactions on Systems, Man, and Cybernetics 34(2), 1108–1116 (2004)
3. Liu, W., Shi, H., Wang, L.: Minimizing Interference in Satellite Communications Using Chaotic Neural Networks. In: ICNC 2007 (2007)
4. Mizuikjz, T.: Optimization of Frequency Assignment. IEEE Transactions on Communications 37(10), 1031–1041 (1989)
5. Cheeneebash, J., Lozano, J.A., Rughooputh, H.C.S.: A Survey on the Algorithms Used to Solve the Channel Assignment Problem. Recent Patents on Telecommunicatione 1(1), 54–71 (2012)
6. Kunz, D.: Channel assignment for cellular radio using neural networks. IEEE Transactions on Vehicular Technology 40(1), 188–193 (1991)
7. Wang, J., Cai, Y., Yin, J.: Multi-start stochastic competitive Hopfield neural network for frequency assignment problem in satellite communications. Expert Systems with Applications 38(1), 131–145 (2011)
8. Zhang, C., Ouyang, D., Ning, J.: An artificial bee colony approach for clustering. Expert Systems with Applications 37(7), 4761–4767 (2010)
9. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report-TR06 (October 2005)
10. Funabiki, N., Nishikawa, S.: A gradual neural-network approach for frequency assignment in satellite communication systems. IEEE Transactions on Neural Networks 8(6), 1359–1370 (1997)

Capturing Geographical Influence in POI Recommendations

Shenglin Zhao[1,2], Irwin King[1,2], and Michael R. Lyu[1,2]

[1] Shenzhen Research Institute
The Chinese University of Hong Kong, Shenzhen, China
[2] Department of Computer Science & Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
{slzhao,king,lyu}@cse.cuhk.edu.hk

Abstract. Point-of-Interest (POI) recommendation is a significant service for location-based social networks (LBSNs). It recommends new places such as clubs, restaurants, and coffee bars to users. Whether recommended locations meet users' interests depends on three factors: user preference, social influence, and geographical influence. Hence extracting the information from users' check-in records is the key to POI recommendation in LBSNs. Capturing user preference and social influence is relatively easy since it is analogical to the methods in a movie recommender system. However, it is a new topic to capture geographical influence. Previous studies indicate that check-in locations disperse around several centers and we are able to employ Gaussian distribution based models to approximate users' check-in behaviors. Yet centers discovering methods are dissatisfactory. In this paper, we propose two models—Gaussian mixture model (GMM) and genetic algorithm based Gaussian mixture model (GA-GMM) to capture geographical influence. More specifically, we exploit GMM to automatically learn users' activity centers; further we utilize GA-GMM to improve GMM by eliminating outliers. Experimental results on a real-world LBSN dataset show that GMM beats several popular geographical capturing models in terms of POI recommendation, while GA-GMM excludes the effect of outliers and enhances GMM.

Keywords: Gaussian Mixture Model, Genetic Algorithm, Geographical Influence, Point-of-Interest, Location Recommendation.

1 Introduction

Point-of-interest (POI) recommendation is a significant service for location-based social networks (LBSNs). With the development of mobile devices and Web 2.0 technologies, many LBSNs like Foursquare and Gowalla emerge and attract many users. These LBSNs allow users to check in their locations, make friends, and share location-related information. In order to help users discover new interesting places in LBSNs, POI recommendations arise.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 530–537, 2013.

How to recommend a point of interest? User preference, social influence, and geographical influence are three aspects responsible for users' check-in activities [14,15]. Generally we derive user preference from user-based collaborative filtering, explore social influence based on users' social relationships, and model geographical influence from check-in locations' spacial features. And then we construct a POI recommendation system in the way of combining those three kinds of influence. The representative work is as follows. Ye et al. [15] propose a linear fused framework to combine them and Cheng et al. [2] propose a fused model to recommend a point of interest.

For POI recommendation in LBSNs, research about geographical influence is new and requires more attention, comparing with user preference and social influence. It is well-defined on how to derive user preference and social influence in a recommendation system [7,11]. Note that users' evaluations for items reflect their preferences and friends are inclined to share preferences. We derive user preference from user-based collaborative filtering and introduce social influence by containing similarity among friends. For POI recommendation system, we use collaborative filtering method to get user preference through treating location as item and check-in frequency as rating value, and we capture social influence by including friends' similarity in check-in locations [2,14,15,1]. In 2010, Ye et al. [14] first propose POI recommendation for LBSNs and utilize power law principle to model users' geographical influence. It is similar to a Gaussian model. Cho et al. [3] study user movement in LBSNs inspired by Gonzalez's discovery [4]. The study focuses on those users who frequently check in, since Gonzalez's discovery bases on call logs data that have strong periodic property. They propose a periodic mobility model (PMM) to capture user's geographical influence for location prediction in LBSNs. Experimental data select users whose check-in records are more than 10 times one day. Cheng et al. [2] propose multi-center Gaussian model (MGM) to capture geographical influence. This model assumes a user's check-in locations disperse around several centers and utilizes a greedy method to discover centers. It defines a district by a fixed distance and thus ignores discrepancy between users. How to capture geographical influence? Gaussian distribution based models perform well in previous studies but we still encounter problems in discovering centers accurately and eliminating the effect of outliers.

To find activity centers more accurately and eliminate outliers, we propose two models—Gaussian mixture model (GMM) and genetic algorithm based Gaussian mixture model (GA-GMM) to capture geographical influence. In geographical perspective, whether one checks in depends on the locations' transport convenience—people prefer places that are nearer to their activity centers. Those frequent check-in places naturally form one's activity district. According to location's spacial clustering feature, we apply GMM to find one's activity district centers. However, outliers exist in the observed data that do harm to learn the model. How to eliminate the impact of outliers? Thang et al. [12] propose a genetic algorithm based EM algorithm to implement the trimmed likelihood estimate (TLE) method [10] to eliminate the outliers in mixture models.

We exploit this genetic based EM algorithm to train GMM. The genetic algorithm based GMM (GA-GMM) improves GMM and finds user's activity centers more accurately.

Our contributions are as follows. First, we propose GMM to automatically learn users' activity centers via exploring their check-in history records. Moreover, we enhance GMM by GA-GMM to intelligently eliminate outliers. Finally, we conduct experiments on a real-world LBSN dataset and demonstrate that the proposed models capture the geographical information better and improve the accuracy of POI recommendation.

The remainder is organized as follows. Section 2 introduces the related work. Section 3 demonstrates the two models: GMM and GA-GMM. Section 4 compares experimental results of different models. In the end, section 5 summarizes and outlines further work.

2 Related Work

In this part, we introduce related work in three aspects: POI recommendation in LBSNs, geographical influence capturing methods, and GA-GMM.

POI recommendation in LBSNs is a new research topic. POI recommendation is widely used in GPS-based mobile devices at first [5,6]. In 2010 Ye et al. [14] first propose POI recommendation in LBSNs. Further Ye et al. [15] point out that user preference, social influence, and geographical influence are three aspects responsible for recommending a point of interest and among them geographical influence is the most important. The representative work is as follows. Ye et al. recommend a point of interest through a linear fused framework combining user preference, social influence, and geographical influence [15]. Cheng et al. [2] propose a fused model to combine them to recommend a point of interest.

Study of geographical influence capturing methods is new for POI recommendation. In 2010 Ye et al. [14] first propose POI recommendation for LBSNs and arise a power law principle to capture geographical influence for POI recommendation. Earlier related work about geographical influence appears in the study of user movement pattern. Gonzalez et al. [4] build a model using call logs and discover that activities of an individual usually center around a small number of frequently visited locations. Based on this, Cho et al. [3] study the specific users frequently checking in and propose a periodic mobility model (PMM) to capture geographical influence for location prediction in LBSNs. Cheng et al. [2] employ multi-center Gaussian model (MGM) to capture the geographical feature of locations in the proposed fused POI recommendation model.

Genetic algorithm based GMM (GA-GMM) is a method to eliminate outliers when learning GMM. Trimmed likelihood estimate (TLE) method is adopted to eliminate outliers in some studies of mixture model analysis [10]. Thang et al. [12] first propose a genetic algorithm based method to implement the trimmed likelihood estimate method to train mixture models and demonstrate the performance through a genetic algorithm based GMM (GA-GMM). Wang et al. utilize the GA-GMM to process EEG signal and apply it on brain-computer interface [13].

3 Models

3.1 Gaussian Mixture Model (GMM)

Gaussian mixture model (GMM) [9] is the most widely used mixture model. We can formulize it as follows:

$$p(\mathrm{x}_i) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathrm{x}_i|\mu_k, \sum_k),$$

where $p(\mathrm{x}_i)$ denotes probability dense distribution of data x_i, μ_k indicates mean value, $\sum_k$ indicates covariance matrix for a base distribution, K denotes the number of base components, and π_k is the mixing coefficient.

We exploit GMM to capture geographical influence in POI recommendation. Each Gaussian distribution component represents an activity district and the mean value denotes the longitude and latitude of the district center. Centers may be his home, office, or some specific entertainment place. We assume places nearer to some center are geographically easier to arrive and people prefer those places.

How to recommend a point of interest through GMM? For a user, a location's geographical information ([longitude, latitude]) in his check-in history records represents data x_i. We recommend POIs through the following steps:

1. Learn the parameters of GMM,
2. Calculate candidate locations' probabilities fitting the trained model, and
3. Sort the candidate locations and recommend the top K locations.

3.2 Genetic Algorithm Based Gaussian Mixture Model (GA-GMM)

In order to eliminate the effect of outliers, we introduce a genetic algorithm based Gaussian mixture model (GA-GMM). Generally we could use maximum likelihood EM algorithm to learn GMM [9]. If we use Θ to denote the parameters, likelihood function could be represented as

$$p(X|\Theta)_{ML} = \prod_{i=1}^{n} p(\mathrm{x}_i|\Theta).$$

Further, if we use the logarithm form, we can denote the objective of maximum likelihood EM algorithm as follows:

$$\hat{\Theta}_{ML} = \arg\max \log p(X|\Theta)_{ML} = \arg\max \sum_{i=1}^{n} \log p(\mathrm{x}_i|\Theta). \tag{1}$$

This formula includes all observed data. Trimmed likelihood estimate (TLE)—that aims to to select the subset of data with maximum sum of likelihood values—is used to eliminate the outliers [10]. We can use a genetic algorithm to find the optimal subset and exploit maximum likelihood EM algorithm to

learn the parameters of GMM, as illustrated in Algorithm 1 [12]. In this case, the objective function could be represented as

$$\log p_{TLE}(X|\Theta) = \sum_{i=1}^{n} w_i \log p(\mathrm{x}_i|\Theta), \tag{2}$$

where $\forall i = 1, 2, ..., n, w_i \in \{0, 1\}$ and $\sum_{i=1}^{n} w_i = m$, m represents the number of valid data. When $w_i = 1$, it indicates that the corresponding data is chosen into the subset. Otherwise, the data is an outlier and should be discarded. Hence, the result is a subset of size m out of n original samples, which fits GMM most in terms of likelihood contribution.

As a genetic algorithm, GA-GMM contains properties of genetic algorithm—it includes encoding scheme, fitness function, and operators like crossover, mutation, and selection. We use the standard way to implement crossover and selection [8]. Encoding scheme, fitness function, and a self-defined mutation (Guided Mutation) are defined as follows.

Definition 1. *Encoding scheme. The chromosome is encoded into a binary string and each bit represents the existence of corresponding observed data. Each chromosome and its corresponding mixture model will be a possible solution to our problem.*

Definition 2. *Fitness function. The fitness score function is set as the trimmed logarithm likelihood of the corresponding GMM of a chromosome*—$\log p_{TLE}(X|\Theta)$.

Definition 3. *Guided Mutation. Guided Mutation ensures the chromosome in a population to mutate toward maximizing fitness score. It means we choose chromosome that has higher value fitting trained GMM.*

4 Experiment

4.1 Setup and Metrics

We prepare the data by cleaning and splitting. We filter locations of less than 10 visits. And then we split the dataset into three non-overlapping sets in sequence: a redundant set, a training set, and a test set. The test set keeps 10% of the whole data set. We test different cases in which the proportion of training data is 90% and 50% respectively. When training data set is 90%, there is no redundant data. When the training data set is 50%, redundant data is the former 40% data that will be discarded.

We evaluate the performance of different models in capturing geographical influence by the accuracy of POI recommendation that is measured by Precision and Recall. POI recommendation is to recommend the top-N highest ranked locations. However, the system should not recommend locations user has checked in. To evaluate the performance of POI recommendation, we use the Precision@N and Recall@N as the metrics that are standard metrics to measure the performance of POI recommendation [15]. Precision@N defines the ratio of recovered POIs to the N recommended POIs and Recall@N defines the ratio of recovered POIs to the size of test set.

Algorithm 1. Genetic-based Expectation Maximization Algorithm

1. t=0;
2. Initialize $P_0(t)$;
3. **for** $t = 1 : G$ **do**
4. $P_1(t) \leftarrow$ perform several cycles of EM on $P_0(t)$;
5. $P_2(t) \leftarrow$ Guided Mutation in $P_1(t)$;
6. $fScore_2 \leftarrow$ evaluate $P_2(t)$;
7. $P_0(t)^{'} \leftarrow$ selection and crossover to generate offspring from $P_2(t)$;
8. $P_1(t)^{'} \leftarrow$ perform several cycles of EM on $P_0(t)^{'}$;
9. $P_2(t)^{'} \leftarrow$ Guided Mutation in $P_1(t)^{'}$;
10. $fScore_2^{'} \leftarrow$ evaluate $P_2(t)^{'}$;
11. $P_3(t) \leftarrow$ selection from $[P_2(t), P_2(t)^{'}]$;
12. $iBest \leftarrow$ best individual from $P_3(t)$;
13. **if** $iBest$ satisfies convergence condition **then** break;
14. $P_0(t+1) \leftarrow P_3(t)$;
15. $t = t + 1$;
16. Perform EM on $iBest$ until convergence;

4.2 Dataset

We use the Gowalla data records from February 2009 to September 2011. We select 3836 active users' records to experiment. We define active users as users whose check-ins are more than 1000 times and experience of using Gowalla is more than 1 year. After removing locations with less than 10 visits, all check-ins of active users include 183,667 different locations. We illustrate statistics of the data in Table 1, where "C." represents the check-in times of a user and "T." represents the time span (unit is day) from first check-in to last check-in.

Table 1. Data statistics

Min. C.	Max. C.	Avg. C.	Min. T.	Max. T.	Avg. T.
1,001	50,243	2,505	366	968	593

4.3 Results

We compare the POI recommendation performance of GMM and GA-GMM with Gaussian model (GM) and Multi-center of Gaussian model (MGM) [2] when training data set is 90% and 50% respectively.

Gaussian model (GM) [4] is a baseline model used in [3]. It models human movement as a stochastic process centered around a single point.

Multi-center Gaussian model (MGM) [2] is a latest model. It uses a fixed distance to define a district. When check-ins in a district are more than a threshold, the mean of all check-ins is the center. It utilizes a greedy method to find the district and requires no overlapping between two districts.

We illustrate experimental results in Fig. 1. GMM outperforms GM and MGM; further GA-GMM improves GMM. Hence, GA-GMM could better capture the geographical influence. In the experiment we set the number of centers in GMM and GA-GMM as 2 for simplicity, since Cho et al. pose that the check-in behaviour comprises two states in [3]. We set the radius of region in MGM as 1 kilometer and the threshold as 10% (that means the ratio of check-ins in one district is at least 10% of all his check-ins).

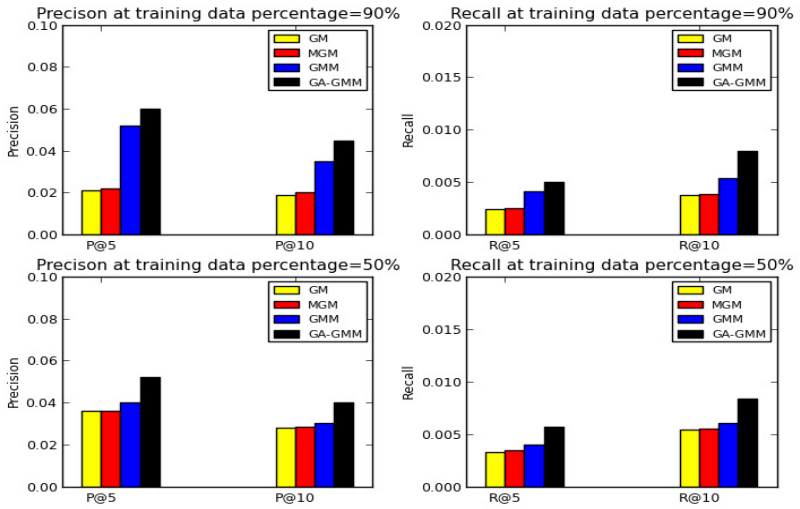


Fig. 1. Comparison of different models

5 Conclusions and Further Work

We apply GMM and GA-GMM to capture geographical influence in POI recommendation. According to experimental results, we draw conclusions as follows. 1) GMM outperforms the baseline model GM and the latest model MGM. 2) GA-GMM eliminates the outliers of data and improves GMM. It discovers the activity centers more precisely, which increases the accuracy of POI recommendation.

There are two aspects of further work. One is to establish a sophisticated POI recommendation system through adding the influence of user preference and social relationship. In this paper, we focus on how to model the geographical feature of check-in activities. For a POI recommendation system, we still have to consider more features of LBSNs such as user preference and social relationship. The other is to improve the efficiency. We learn each user's model separately. That provides opportunities to implement a parallel version.

Acknowledgments. The work described in this paper was partially supported by the Shenzhen Major Basic Research Program (Project No. JC201104220300A) and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. CUHK413212, CUHK415212).

References

1. Cheng, C., Yang, H., Lyu, M.R., King, I.: Where you like to go next: Successive point-of-interest recommendation. In: IJCAI, Beijing, China (2013)
2. Cheng, C., Yang, H., King, I., Lyu, M.R.: Fused matrix factorization with geographical and social influence in location-based social networks. In: AAAI 2012: Proceedings of Twenty-Sixth Conference on Artificial Intelligence (2012)
3. Cho, E., Myers, S.A., Leskovec, J.: Friendship and mobility: user movement in location-based social networks. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1082–1090. ACM (2011)
4. Gonzalez, M.C., Hidalgo, C.A., Barabasi, A.L.: Understanding individual human mobility patterns. Nature 453(7196), 779–782 (2008)
5. Horozov, T., Narasimhan, N., Vasudevan, V.: Using location for personalized poi recommendations in mobile environments. In: International Symposium on Applications and the Internet, SAINT 2006, p. 6. IEEE (2006)
6. Kang, E.-y., Kim, H., Cho, J.: Personalization method for tourist point of interest (POI) recommendation. In: Gabrys, B., Howlett, R.J., Jain, L.C. (eds.) KES 2006. LNCS (LNAI), vol. 4251, pp. 392–400. Springer, Heidelberg (2006)
7. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42(8), 30–37 (2009)
8. Melanie, M.: An introduction to genetic algorithms, Cambridge, Massachusetts London, England, Fifth printing, vol. 3 (1999)
9. Murphy, K.P.: Machine learning: a probabilistic perspective. The MIT Press (2012)
10. Neykov, N., Filzmoser, P., Dimova, R., Neytchev, P.: Robust fitting of mixtures using the trimmed likelihood estimator. Computational Statistics & Data Analysis 52(1), 299–308 (2007)
11. Ricci, F., Shapira, B.: Recommender systems handbook. Springer (2011)
12. Thang, N.D., Lihui, C., Keong, C.C.: An outlier-aware data clustering algorithm in mixture models. In: 7th International Conference on Information, Communications and Signal Processing, ICICS 2009, pp. 1–5. IEEE (2009)
13. Wang, B., Wong, C.M., Wan, F., Mak, P.U., Mak, P.I., Vai, M.I.: Gaussian mixture model based on genetic algorithm for brain-computer interface. In: 2010 3rd International Congress on Image and Signal Processing (CISP), vol. 9, pp. 4079–4083. IEEE (2010)
14. Ye, M., Yin, P., Lee, W.C.: Location recommendation for location-based social networks. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 458–461. ACM (2010)
15. Ye, M., Yin, P., Lee, W.C., Lee, D.L.: Exploiting geographical influence for collaborative point-of-interest recommendation. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 325–334. ACM (2011)

An Efficient Clustering Method for Massive Dataset Based on DC Programming and DCA Approach

Ta Minh Thuy, Hoai An Le Thi, and Lydia Boudjeloud-Assala

Laboratory of Theoretical and Applied Computer Science (LITA)
UFR MIM, University of Lorraine, Ile du Saulcy, 57045 Metz, France
minh-thuy.ta5@etu.univ-lorraine.fr,
{hoai-an.le-thi,lydia.boudjeloud-assala}@univ-lorraine.fr

Abstract. In this paper, we study an efficient nonconvex optimization method for clustering on massive datasets. Our approach consists of two phases and is based on DC (Difference of Convex functions) programming and DCA (DC Algorithms). In the first phase, the data is divided into subsets on which an efficient DCA for clustering is investigated. In the second phase, another DCA for weighted clustering on the set of centers obtained by phase 1 is presented. The numerical results on real datasets show the efficiency of our method.

Keywords: Clustering, Data stream, Massive dataset, DCA, DCA Weight.

1 Introduction

Clustering on massive datasets is a challenging research area in data mining. The data massive will be linked not only to the volume, but also the complexity of processing. The technical data clustering and learning must adapt to such problems.

Nowadays, various massive data exists in the form of data streams. It concerns applications such as telephone records, banking, multimedia data,... For these applications, data stream is an appropriate model when a large volume of data is arriving continuously and it is either unnecessary or impractical to store the data in some form of memory [3]. In this context, we have focused on the approaches of data processing as an optimization problem and applied with data streams.

Nittel et al. [9] introduced a *partial/merge* method which consists of 2 phases for problem clustering mass of data. In the phase 1, data is subdivided into p subsets and one performs an algorithm clustering on each subset, and the classical K-Means is used. In the phase 2, a K-Means Weight based algorithm is applied on the set of weight centers obtained from phase 1. The geoscientific datasets have been tested. Since two problems above are nonconvex optimization problems, so our approach explore an optimization nonconvex technique based on DC programming (Difference of Convex functions) and DCA (DC Algorithm)

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 538–545, 2013.

[8]. We use a DCA clustering algorithm instead of K-Means classical in the first phase. Subsequently, we developed another DCA clustering algorithm for second phase. Comparative experiments with clustering K-Means Weight approach [9] on different datasets are reported.

The remaining paper is structured as follows: section 2 introduces DC programming and DCA. Section 3 introduces DCA clustering algorithms. Numerical results on real data sets and some remarks will be presented in section 4. Concludes the paper and future works in last section.

2 DC Programming and DCA

In this paper we study a clustering algorithm in the nonconvex programming framework based on DC programming and DC Algorithm, which were introduced by Pham Dinh Tao in their preliminary form in 1985 [8]. They have been extensively developed since 1994 by Le Thi Hoai An and Pham Dinh Tao and now, DCA has been successfully applied to a lot of various large-scale (smooth or non-smooth) nonconvex programs [6], [7], [8].
General DC program as:

$$\alpha = \inf\{f(x) := g(x) - h(x) \ : \ x \in \mathbb{R}^n\} \quad (P_{dc})$$

where g, h are lower semi-continuous proper convex functions on $\mathbb{R}^n$. The idea of DCA is simple: each iteration ℓ of DCA approximates the concave part $-h$ by its affine majorization (that corresponds to taking $y^\ell \in \partial h(x^\ell)$) and minimizes the resulting convex function (that is equivalent to determining a point $x^{\ell+1} \in \partial g^*(y^\ell)$ with g^* is the conjugate function of the convex function g).

The generic DCA schema is shown below.
DCA Schema:
Let $x^0 \in \mathbb{R}^n$ be an initial guess. Set $k := 0$
REPEAT
$\gamma^k \in \partial H(x^k)$. $x^{k+1} \in \operatorname{argmin}\{G(x) - \langle x, \gamma^k \rangle : x \in \mathbb{R}^n\}$.
Set $k + 1 \leftarrow k$
UNTIL $\|x^{k+1} - x^k\| \leq \epsilon(|x^k| + 1)$ or $|f(x^{k+1}) - f(x^k)| \leq \epsilon(|f(x^{(k)})| + 1)$.
DCA schema have the following properties ([7], [8]):

i) the sequence $\{g(x^k) - h(x^k)\}$ is decreasing,
ii) if the optimal value α of problem (P_{dc}) is finite and the infinite sequences $\{x^k\}$ is bounded, then every limit point $\widetilde{x}$ of the sequence $\{x^k\}$ is a critical point of $g - h$.

For instant, it is worth to note that the construction of DCA involves the convex DC components g and h but not the DC function f itself. Moreover, a DC function f has infinitely many DC decompositions $g - h$ which have a crucial impact on the qualities (speed of convergence, robustness, efficiency, globality of computed solutions,...) of DCA. The solution of a nonconvex program by DCA must be composed of two stages: the search of an appropriate DC decomposition and that of a good initial point.

3 DCA Clustering Algorithms

3.1 DCA for the First Phase

An instance of the partitional clustering problem consists of a data set $\mathcal{A} := \{a^1, ..., a^q\}$ of q points in $\mathbb{R}^n$, a measured distance, and an integer k; we are to choose k members $x^\ell \in \mathbb{R}^n (\ell = 1, ..., k)$ as "centroid" and assign each member of $\mathcal{A}$ to its closest centroid. The assignment distance of a point $a \in \mathcal{A}$ is the distance from a to the centroid to which it is assigned, and the objective function, which is to be minimized, is the sum of assignment distances. The corresponding optimization formulation is expressed as: ($\|.\|$ denotes the Euclidean norm)

$$\min \quad \left\{ \sum_{i=1}^{q} \min_{\ell=1,...,k} \left\| x^\ell - a^i \right\|^2 : x^\ell \in \mathbb{R}^n, \ell = 1, \ldots, k \right\}. \tag{1}$$

The DCA applied to problem (1) has been developed in [6]. We will give below a brief description of this algorithm:

DCA for Solving the Problem (1):

Initialization: Let $\epsilon > 0$ be given, $X^{(0)}$ be an initial point in $\mathbb{R}^{k \times n}, p := 0$;

Repeat:

Calculate $Y^{(p)} \in \partial H(X^{(p)})$ by: $Y^{(p)} = qX^{(p)} - B - \sum_{i=1}^{q} e_{j(i)}^{[k]} (X_{j(i)}^{(p)} - a^i)$

Calculate $X^{(p+1)}$ according to: $X^{(p+1)} := (B + Y^{(p)})/q$.

Set $p = p + 1$.

Until: convergence of $\{X^{(p)}\}$.

where $B \in \mathbb{R}^{k \times n}$, $B_\ell := \sum_{i=1}^{q} a^i$ and $e_j^{[k]}$ being the canonical basis of $\mathbb{R}^k$.

3.2 DCA for Second Phase

In this section, we developed a DC decomposition for problem clustering on the set of weight centers, the DC Algorithm for this problem called by DCA Weight.

Denote $b^i (i = 1, ..., s = k * p)$ is the set of centers obtained from phase 1. Each b^i is weighted by w_i which is the number of points that were assigned to the centers b^i. In the second phase, we are to find again k members u^ℓ as "centroid" of the set b^i, then the corresponding optimization formulation is expressed as:

$$\min \quad \left\{ \sum_{i=1}^{s} w_i * \min_{\ell=1,...,k} \left\| u^\ell - b^i \right\|^2 : u^\ell \in \mathbb{R}^n, \ell = 1, \ldots, k \right\}. \tag{2}$$

To simplify related computations in DCA for solving problem (2) we will work on the vector space $\mathbb{R}^{k \times n}$ of $(k \times n)$ real matrices. We have:

$$\begin{aligned} F(U) &= \frac{1}{2} \sum_{i=1}^{s} w_i * \min_{\ell=1,..,k} \|U_\ell - b^i\|^2 \\ &= \sum_{i=1}^{s} \sum_{\ell=1}^{k} w_i * \frac{1}{2} \|U_\ell - b^i\|^2 - \sum_{i=1}^{s} w_i * \max_{j=1,..,k} \frac{1}{2} \sum_{\ell=1, \ell \neq j}^{k} \|U_\ell - b^i\|^2 \\ &= G(U) - H(U). \end{aligned} \tag{3}$$

where $G(U) = \sum_{i=1}^{s}\sum_{\ell=1}^{k} G_{i\ell}(U), G_{i\ell}(U) = \frac{1}{2}\ w_i\|U_\ell - b^i\|^2$ and $H(U) = \sum_{i=1}^{s} H_i(U)$, $H_i(U) = \max\limits_{j=1,..,k} H_{ij}(U) = \max\limits_{j=1,..,k} \sum_{\ell=1,\ell\neq j}^{k} \frac{1}{2}\ w_i\|U_\ell - b^i\|^2$. Since $G(U)$ and $H(U)$ are convex functions, problem (2) is a DC program.

Recall that m is the objects in whole data set, w_i denote the number of points that were assigned to the centers b^i, $s = k * p$ with p is number of subsets, so $\sum_{i=1}^{s} w_i = m$. It leads to:

$$G(U) = \frac{1}{2}\sum_{i=1}^{s}\sum_{\ell=1}^{k} w_i * \|U_\ell - b^i\|^2 = \frac{m}{2}\|U\|^2 - \langle B, U\rangle + \frac{1}{2}\|A\|^2 \tag{4}$$

where $A_i = \sqrt{w_i}b^i \in \mathbb{R}^n$ (i=1,..., s); $B_\ell = \sum_{i=1}^{s} w_i b^i \in \mathbb{R}^n (\ell = 1, ..., k)$. The DC program (3) then is minimizing the difference of the simplest convex quadratic function $G(U)$ and the nonsmooth convex function one $H(U)$. This nice feature is very convenient for applying DCA, which consists of solving a sequence of approximate convex quadratic programs whose solutions are explicit.

a) DCA for Solving the Problem (2)

According to the description of DCA, the DCA schema applied to (3) amounts to computing the two sequences $\{U^{(p)}\}$ and $\{V^{(p)}\}$ in $\mathbb{R}^{k\times n}$ such that: $V^{(p)} \in \partial H(U^{(p)}), U^{(p+1)} \in \partial G^*(V^{(p)})$. First, we express the convex function H_{ij} by:

$$\begin{aligned} H_{ij}(U) &= \sum_{\ell=1,\ell\neq j}^{k} \frac{1}{2} w_i\|U_\ell - b^i\|^2 = \sum_{\ell=1}^{k} \frac{1}{2} w_i\|U_\ell - b^i\|^2 - \frac{1}{2} w_i\|U_j - b^i\|^2 \\ &= \frac{1}{2} w_i\|U - B^{[i]}\|^2 - \frac{1}{2} w_i\|U_j - b^i\|^2 \end{aligned}$$

where $B^{[i]} \in \mathbb{R}^{k\times n}$ is the matrix whose rows are all equal to b^i.

Therefore, $\nabla H_{ij}(U) = w_i(U - B^{[i]}) - e_j^{[k]} w_i(U_j - b^i)$ with $e_j^{[k]} : j = 1, ..., k$ being the canonical basis of $\mathbb{R}^k$. Hence, we get the following simpler matrix formula for computing ∂H: $V \in \partial H(U) \Leftrightarrow V = \sum_{i=1}^{s} V^{[i]}$ with $V^{[i]} \in \partial H_i(U)$ where $V^{[i]}$ is a convex combination of $\{\nabla H_{ij}(U) : j \in K_i(U)\}$, i.e:

$$V^{[i]} = \sum_{j\in K_i(U)} \lambda_j^{[i]} \nabla H_{ij}(U) \text{ with } \lambda_j^{[i]} \geq 0 \text{ for } j \in K_i(U) \text{ and } \sum_{j\in K_i(U)} \lambda_j^{[i]} = 1,$$

with $K_i(U) := \{j = 1, \ldots, k : H_{ij}(U) = H_i(U)\}$. In particular, we can take for $i = 1, ..., s : V^{[i]} = w_i(U - B^{[i]}) - \sum_{j\in K_i(U)} \lambda_j^{[i]} e_j^{[k]} w_i(U_j - b^i)$ and the corresponding $V \in \partial H(U)$ is defined by:

$$V = mU - \sum_{i=1}^{s} w_i B^{[i]} - \sum_{i=1}^{s} e_{j(i)}^{[k]} w_i(U_j - b^i). \tag{5}$$

Now, we focus on computing U by DCA schema. As the function G is strictly convex quadratic and its conjugate G^* is differentiable, from (4) we have:

$U = \nabla G^*(V)$. So, $V = \nabla G(U) = mU - B$. Finally:

$$U = (V + B)/m. \tag{6}$$

b) Description of DCA to solve problem (2)

Initialization: Let $\epsilon > 0$ be given, $p := 0$. $U^{(0)}$: initial cluster centers.
Repeat:
Calculate $V^{(p)}$ from (5).
Calculate $U^{(p+1)}$ from (6).
Set $p = p + 1$.
Until: $|F^{(p+1)} - F^{(p)}| \leq \epsilon(|F^{(p)}| + 1)$ or $\|U^{(p+1)} - U^{(p)}\| \leq \epsilon(\|U^{(p)}\| + 1)$.

This algorithm has convergence properties of general DC programs [7], [8].

4 Experiments

a) Datasets
We perform experiments with 7 real-world datasets: *Forest Covertype (FCT)* taken from [1], *KDD98* [5], *KDD99*, *HyperPlane(HP)* from [13], *SEA*[10], Sensor_1[12] and Sensor_2 from [11]. The information is summarized in table 1.

KDD98 contains 95.412 records of information about people who have made charitable donations. This dataset is converted into a data stream with 56 attributes, 10 classes (as [5]). *KDD-CUP'99* corresponds to a challenging problem for dynamic stream clustering. Dataset contains 494.021 points, 41 attributes with 24 types attack. We only use 34 continuous attributes (as [4]) and the type attacks fall into two main category: normal and abnormal. *Sensor_1* holds five information: rcd_minutes, temperature, humidity, light and voltage that selected from four regions and the processed stream has 1.169.260 points. *Sensor_2* contains 2.219.803 points, this experiment aims to infer the illuminance state based on the measurements provided by each sensor, illuminance higher than 200 are considered as class 1, otherwise considered as class -1. All datasets are normalized before experiments.

Table 1. Datasets

Dataset	No. Points	No. Att	No. Clusters	No. Subsets	No. Elements/Subset
SEA	60000	3	2	10	6.000
HP	100000	10	5	10	10.000
KDD98	95412	56	10	10	9.541
KDD99	494021	34	2	10	49.402
FCT	581012	54	7	10	58.101
SENSOR_1	1169260	5	4	10	116.926
SENSOR_2	2219803	5	2	10	221.980

b) Testing Protocols
We perform clustering on each subset by two algorithms DCA and K-Means with the same initial points, which are randomly chosen from given objects. Then, two algorithms DCA Weight and K-Means Weight are applied on the set of centers obtained. Initial points are chosen from s center points with k largest weights. Since the object function is sensitive to the initial seeds, we have run each algorithm with 5 different sets of initial seeds. By investigating regrouping of data after getting the centers of dataset, we study 2 approaches:

Option 1: if the data scanned only once, the elements are assigned to cluster corresponding to their center in the phase 1.
Option 2: the data can be scanned in the subset more than one time. Then, the elements are assigned to the closest center which obtained in phase 2.

c) Evaluation
In our study, we focus on 3 criteria: the value of objective function, the CH value ([2]) and CPU running time.

Objective Function: The objective function of method with 2 phases is sum of the assignment distances and defined as, with c_j are centers data in step 2 :

$$F = \sum_{j=1}^{k} \sum_{x_i \in C_j} w_i * \parallel x_i - c_j \parallel^2; \quad i = 1, ..., s$$

CH Value (Calinski RB, Harabasz): The value cluster index of Calinski and Harabasz [2] describe the values of between-cluster and within-cluster, where *traceB* denotes the sum of squares between different clusters and *traceW* is the squared differences of all objects in a cluster from their respective cluster center.

$$CH(K) := \frac{[traceB/(k-1)]}{[traceW/(m-k)]}$$

d) Numerical Results
All algorithms have been implemented in the VisualC++2008, run on a PC Intel i5CPU650, 3.2 GHz of 4GB RAM. From numerical results, we observe that:

i) DCA in general is better than the one based on K-Means. DCA Weight gives better objective function on all experiments. The CH values of DCA is better than the CH value of K-Means on 6/7 datasets with the option 1; on all datasets with the option 2.
ii) In the set of datasets *SEA*, *KDD99*, *FCT*, *Sensor_1*: DCA-Weight not only gives better solutions in term of CH values and objective function, but also on the standard deviation (SD) values.
iii) DCA provides good solutions in reasonable time.
iv) The execution time of option 1 is faster than option 2, but the result of CH values of option 2 is better than option 1 in all experiments.
v) The method explores parallel process by multi processer or multi-machines in the first step.

Table 2. The objective functions of DCA (1) and K-Means (2)

Datasets	Min Obj.		Max. Obj		Avg. Obj		SD	
Name	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)
SEA	**155606**	180179	**189572**	219446	**176025**	195418	**11278**	16205
KDD98	**19307**	25551	**42024**	45091	**29205**	32097	7528	**6859**
HP	**39058**	42588	**41061**	44968	**40266**	43404	**700**	833
KDD99	**78732**	97846	**112099**	113118	**90054**	105460	11521	**5883**
FCT	**172924**	199414	**214908**	275989	**190057**	225873	**18660**	26875
Sensor_1	**40215**	52755	**87710**	93620	**54792**	68986	**17097**	19734
Sensor_2	**74318**	89156	**94469**	108838	**86837**	96987	8377	**6463**

Table 3. The CH values of DCA (1) and K-Means (2) with option 1

Datasets	Min.CH		Max.CH		Avg.CH		SD	
Name	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)
SEA	**7074**	5227	**8748**	7538	**7701**	6655	**600**	960
KDD98	**5813**	5648	**8118**	7719	**7083**	6963	764	**758**
HP	**1466**	1387	1571	**1599**	1510	**1516**	**37**	71
KDD99	**212862**	117160	**326221**	262529	**279700**	199994	**40251**	59966
FCT	**78769**	73023	**96441**	93040	**88739**	85390	**5903**	7338
Sensor_1	**402415**	375917	**875661**	767465	**656922**	594034	**157291**	179894
Sensor_2	**587780**	446899	**807009**	644869	**669221**	550953	84888	**84591**

Table 4. The CH values of DCA (1) and K-Means (2) with option 2

Datasets	Min.CH		Max.CH		Avg.CH		SD	
Name	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)
SEA	**7921**	6557	**9933**	8552	**8687**	7657	**718**	858
KDD98	**6501**	6292	**9582**	8734	**8287**	7815	1001	**889**
HP	**1689**	1680	**1814**	1802	**1770**	1755	47	**40**
KDD99	**344742**	241201	**368429**	345925	**358484**	303059	**8650**	45627
FCT	**88219**	76833	**104638**	99653	**97823**	92091	**6116**	7955
Sensor_1	**436826**	415603	**1027300**	984041	**812461**	724526	**201102**	249924
Sensor_2	**710050**	550402	**937843**	773868	**799818**	670419	**83689**	93094

Table 5. The CPU Time of DCA (1) and K-Means (2)

Datasets	Avg.Time(Option 1)		SD(Option 1)		Avg.Time(Option 2)		SD(Option 2)	
Name	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)
SEA	3,870	**2,294**	**0,018**	0,030	4,962	**3,336**	0,289	**0,254**
KDD98	165,999	**37,743**	**0,004**	0,024	174,965	**46,500**	**0,383**	0,384
HP	**8,742**	14,201	**0,026**	0,028	**12,064**	17,465	**0,383**	0,483
KDD99	41,383	**30,874**	1,348	**0,616**	64,073	**60,218**	1,551	**1,319**
FCT	295,961	**98,133**	16,477	**3,033**	335,945	**146,997**	16,422	**3,077**
Sensor_1	85,621	**53,305**	5,837	**3,153**	108,533	**95,569**	6,239	**4,623**
Sensor_2	113,530	**81,505**	11,289	**7,207**	155,846	**153,308**	11,438	**8,900**

5 Conclusion and Future Works

We have rigorously studied the DC programming and DCA for clustering on mass of data. The classical K-Mean and K-Mean Weight models were reformulated as DC programs. The effects of DC decompositions are well exploited for obtaining effective algorithms. The numerical results on several real datasets show that DCA is an efficient approach for clustering in large datasets. The performance of DCA suggests us to investigating it for feature selection or clustering based on the relationships of the attributes. Works in these directions are in progress.

References

1. Blackard, J.A., Dean, D.J.: Comparative Accuracies of Artificial Neural Networks and Discriminant Analysis in Predicting Forest Cover Types from Cartographic Variables. Computers and Elec. in Agriculture 24(3), 131–151 (2000)
2. Cali'nski, T., Harabasz, J.: A dendrite method for cluster analysis. Communications in Statistics Simulation and Computation 3(1), 1–27 (1974)
3. Callaghan, L.O., Nina, M., Adam, M., Sudipto, G., Rajeev, M.: Streaming-data algorithms for high-quality clustering. IEEE Computer Society, 685–694 (2002) ISBN 0-7695-1531-2
4. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB 2003, vol. 29, pp. 81–92 (2003)
5. Fredrik, F., James, L., Charles, E.: Scalability for clustering algorithms revisited. ACM SIGKDD Explor. Newsletter 2(1), 51–57 (2000)
6. An, L.T.H., Belghiti, M.T., Tao, P.D.: A new efficient algorithm based on dc programming and dca for clustering. J. of Global Optimization 37(4), 593–608 (2007)
7. An, L.T.H., Tao, P.D.: The dc (difference of convex functions) programming and dca revisited with dc models of real world nonconvex optimization problems. Annals of Operations Research 133(1-4), 23–46 (2005)
8. Tao, P.D., An, L.T.H.: Convex analysis approach to d.c. programming: Theory, algorithms and applications. Acta Mathematica Vietnamica, Dedicated to Professor Hoang Tuy on the Occasion of his 70th Birthday 22(1), 289–355 (1997)
9. Nittel, S., Ncgia, S., Leung, K.T., Braverman, A.: Scaling clustering algorithms for massive data sets using data stream. IEEE Computer Society, 417–428 (2003) ISBN 0-7803-7665-X
10. Street, W.N., Kim, Y.: A streaming ensemble algorithm SEA for large-scale classification. In: Proceedings of the 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 26-29, pp. 377–382 (2001)
11. Rastgoo, M., Lemaitre, G., Rafael Palou, X., Miralles, F., Casale, P.: Pruning adaboost for continuous sensors mining applications. In: Ubiquitous Data Mining Works, 20th Euro. Conf. in Arti. Intel.-ECAI, Montpellier, France, August 27-31, pp. 53–57 (2012)
12. Xingquan, Z., Wei, D., Yu, P.S., Zhang, C.: One-class learning and concept summarization for data streams. Know. Inf. Syst. 28(3), 523–553 (2011)
13. Zhu, X.: Stream data mining repository, `http://cse.fau.edu/xqzhu/stream.html` (accessed on September 2012)

A Knowledge-Based Initial Population Generation in Memetic Algorithm for Protein Structure Prediction

Rumana Nazmul and Madhu Chetty

Gippsland School of Information Technology, Monash University, Australia
{rumana.nazmul,madhu.chetty}@monash.edu

Abstract. Predicting the minimum energy protein structure from its amino acid sequence, even under the rather simplified HP lattice model, continues to be an important and challenging problem in computational biology. In this paper, we propose a novel initial population generation strategy for evolutionary algorithm incorporating domain knowledge based on the concept of maximum hydrophobic core formation for Protein structure prediction (PSP) problem. The proposed technique helps the optimization process to commence with diverse seeds and thereby aids in converging to the global solution quickly. The experimental results, conducted on PSP problem using HP benchmark sequences for 2D square and 3D cubic lattice model, demonstrate that the proposed evolutionary algorithm with new core-based population initialization technique is very effective in improving the optimization process in terms of convergence as well as in achieving the optimal energy.

Keywords: Protein Structure Prediction, Hydrophobic Core, Lattice.

1 Introduction

Proteins are important biological macromolecules that play vital role in living organisms. The protein structure prediction problem (PSP) that can be defined as the problem of finding the native structure of a protein having the lowest possible free energy given its amino acid sequence. To deal with the computation complexity of this long-standing NP-hard problem [1], researches have often been carried out with simplified models [2]. The *hydrophobic-polar* (HP) model [2] is one such abstraction for the PSP that captures the hydrophobic effect as the main driving forces for the formation of protein structure. The hydrophobic effect causes hydrophobic amino acids to minimize contact with water and consequently, the hydrophobic residues tend to aggregate together and form a 'core' in the spatial structure, shielded from the surrounding solvent by hydrophilic amino acids. This type of conformation is stable and has global minimum free energy. More details about the model can be found in [3,4]. However, computational complexity of PSP problem even with the simplified models has motivated the investigation of various approximation methods. The non-deterministic methods to solve PSP include genetic algorithms (GA) [5], memetic algorithms (MA) [6], ant colony optimization (ACO) [7], particle swarm

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 546–553, 2013.

optimization (PSO) [8], estimation of distribution algorithms (EDA) [9]. The heuristic methods based on the hydrophobic core are: Constrained Hydrophobic Core Construction (CHCC) [10], Core-directed chain growth algorithm (CG) [11], hybrid GA [12, 13]. Recently, we have proposed the Hydrophobic Core Directed Protein Structure Prediction (HCD-PSP) [3] that estimates the size of maximum core by scrutinizing the input sequence aided with knowledge about the positioning of residues which are critical for folding, based on their own and neighborhood observations. However, the application of the estimated core sizes are limited only to chain-growth algorithm in [3] and cannot cater to the possibility of occurrence of multiple cores in the optimal solution.

In this paper, we propose a new evolutionary approach for protein structure prediction by incorporating the concept of maximum hydrophobic core estimation technique as prior knowledge to generate initial population in memetic algorithm (MA). Both 2D square and 3D cubic HP lattice models are considered here to demonstrate the concept of implementing hydrophobic core in population initialization and also for studying the effect of the proposed strategy in predicting the structure of protein.

2 Preliminaries

2.1 Basic Definitions

Definition 1: Hydrophobic Core (H-Core). In a protein conformation, by *hydrophobic core* (H-Core), we refer to the set of all lattice points which are occupied by hydrophobic residues only. Here, we consider the H-Core as a stack of Z (=1 in case of 2D square lattice) layers, each having a dimension $(X \times Y)$.

Definition 2: Total H (T_H), Cored-H (C_H), Outside H (O_H). Total H (T_H) is defined as the total number of Hs in the input protein sequence, whereas Cored-H (C_H) denotes the number of Hs inside the H-Core. Outside H (O_H) i.e., $(T_H - C_H)$ is the number of Hs that are allowed to be outside the H-Core.

Definition 3: Classified Residue (Isolated H (I_H), Isolated P (I_P), Associated H (A_H)). In a sequence of length Len, an H residue at the i^{th} position $(1 < i \leq Len - 1)$ is defined as an Isolated H (I_H) if both the $(i-1)^{th}$ and $(i+1)^{th}$ residues are P residues. Isolated P (I_P) is defined in a manner similar to the definition of I_H. The H residues that are connected with an I_P in either of the sides, are called Associated H (A_H). In this paper, we call all these residues as the Classified Residues since they play a very important role in estimating the size of the H-Core. The total number of I_H, I_P, A_H are denoted by T_{I_H}, T_{I_P}, T_{A_H}, respectively.

Definition 4: Classified Location $C_L^{'}$. It has been observed that, Classified Residues apply restrictions while positioning the H residues anywhere inside the H-Core. In a 2D HP lattice, a core of any dimension $(X \times Y)$, has exactly the four corner locations that allows only four I_H residues to be placed. On the other hand, for a 3D HP lattice, all the border locations of the top and bottom layers, can accommodate the I_H residues, whereas in the remaining $Z-2$ intermediate

layers, only four corner locations can place at most four I_H residues (similar to H-Core in 2D HP lattice). We call all of these locations as Classified Locations. In 3D lattice, for a core with dimensions $(X^{'}, Y^{'}, Z^{'})$, the number of such Classified Locations $C_L^{'}$ is (*Border locations of the top and bottom layers*) + (*corner locations of the intermediate layers*)$= (2*(2*(X^{'}+Y^{'})-4))+(4*(Z^{'}-2))$. A conceptual H-Core with the corresponding Classified Locations is shown for 3D cubic lattice model in Fig. 1.

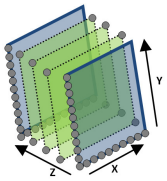


Fig. 1. A conceptual core in 3D HP lattice with possible Classified Locations in various layers. The Classified Locations in the left and down edges for the first and last layers and only 4 possible Classified Locations in the intermediate layers are shown.

2.2 Existing Initial Population Generation Techniques

In all population based methods, initial population play an important role to maintain diversity. Among different techniques for generating initial population, the most widely used approach generates every individual in the population with random moves, followed by verifying the moves with Self-avoiding-walk (SAW) constraints [4, 6]. This technique, although computationally inefficient, is very simple. However, in a recently proposed initial population generation technique DIG (Dynamic Initial-population Generation) [4], an individual is created by sequentially adding moves ensuring that SAW is maintained. Although, this method is faster than the random generation technique, it does not ensure that resulting conformations contain hydrophobic core.

3 Proposed Method

3.1 Core-Based Initial Population Generation

The incorporation of knowledge from the problem domain that can improve the performance of the optimization, can be done either implicitly (i.e., during encoding or constraint optimization) or explicitly during initial population generation [14]. Incorporating prior knowledge in initial population can play an important role because quality seeds can increase the possibility of achieving the global optimal solution through the iterative process of information exchange. In this section, we introduce a novel initial population generation technique called Core-based Initial Population Generation (CIPG) for incorporating knowledge in the population based method (memetic algorithm in our case).

The CIPG algorithm works in two stages. The first stage starts with estimating the Cored-H (C_H), that indicates the number of H residues forming the

maximum possible H-Core by analyzing the $Classified\ Residues$ in the sequence and then calculate the dimensions ($MaxDim$) of the maximum possible H-Core with the aid of the estimated C_H. With the calculated C_H and $MaxDim$, in the second stage, we constitute the set of possible cores of various dimensions with a fraction of C_H and generate the initial population.

Stage-I: (Core Estimation) While calculating the C_H, first we calculate the total number of Hs (T_H) from the given sequence (S). Our assessment on $Classified\ Residues$ illustrates that, these residues cannot be placed everywhere inside the core due to their position as well as the neighborhood in the sequence. According to the definition of associated H (A_H) and the orientation of 2D square or 3D cubic lattice, at least $\lfloor \frac{T_{A_H}}{2} \rfloor$ residues must be placed outside the H-Core. Further, only four of the total I_H residues can be placed inside the core on any of the four corner locations for 2D HP lattice irrespective of the size of the core. Hence, in case of 2D square lattice, we estimate the number of C_H based on the two observations according to $C_H = T_H - max(T_{I_H} - 4, \left\lfloor \frac{T_{A_H}}{2} \right\rfloor)$. Further, the maximum dimension of the approximated core (X_{max},Y_{max}) are: $X_{max} = \sqrt{C_H}, Y_{max} = C_H / X_{max}$ [3]. However, in case of 3D HP lattice, the number of Classified Locations ($C_L^{'}$) varies with the dimensions of the core and thereby C_H is calculated by subtracting only $\lfloor \frac{T_{A_H}}{2} \rfloor$ from T_H, i.e., C_H=T_H-$\lfloor \frac{T_{A_H}}{2} \rfloor$. Further, while calculating the maximum core dimension, we apply a constraint to accommodate the maximum possible I_H residues in the Classified Locations of the core with that particular dimension, as shown in Algorithm 1.

Algorithm 1. MaxCoreDimension3D (C_H, T_{I_H}, $C_L^{'}$)

```
Input: C_H=Maximum possible H for core, T_{I_H}= total number of I_H, C'_L= total
number of Classified Locations
MaxCore = 0
For Z = ∛C_H to 2 do
    For X = √(C_H/Z) to 2 do
        Y = C_H/(X*Z)
        CL' ← CalculateClassLoc(X, Y, Z)
            if C'_L ≥ T_{I_H} and (X*Y*Z) ≥ MaxCore then
                MaxCore = X*Y*Z, MaxDim=(X, Y, Z)
            End If
    End For
End For
Return (MaxCore, MaxDim)
```

Stage-II: (Initial Population Generation with Cores) In this stage, we opt to incorporate the knowledge of the estimated Cored-H (C_H) and dimensions ($MaxDim$) of the maximum possible H-Core in the population initialization of the memetic algorithm. As stated earlier, rather than generating cores with the total C_H for the initial population, we select a smaller percentage of C_H i.e., allowed Cored-H (A_{C_H}) to generate the initial seeds. Further, we generate the set of possible cores of varying dimensions such that for a particular (k^{th}) core size, any of its dimensions does not exceed the $MaxDim$ (i.e., $X_k \leq X_{max}$, $Y_k \leq Y_{max}$ and $Z_k \leq Z_{max}$ in 3D lattice) and the total Hs consisting of the core is less than

or equal to A_{C_H} (i.e., $X_k \times Y_k \times Z_k \leq A_{C_H}$). Let, T_C be the total number of unique core sizes to be generated for a given sequence. However, in our proposed CIPG algorithm, we allow T_{rand} (=RND% of the population) individuals to be generated with random initialization method. Thus, we calculate the number of individuals to be generated for each unique core size as: $I_{core} = (pop - T_{rand})/T_C$. Finally, the following three-step method is applied to generate a conformation for k^{th} core dimension (X_k, Y_k, Z_k), where $1 \leq k \leq T_C$.

Step-1 (FillCore): The method starts with selecting a random starting H ($StartPos$) from first 50% Hs (to ensure sufficient number of Hs are available for generating the core) and then consider a sub-sequence containing S_H number of H residues (where $S_H = X_k \times Y_k \times Z_k$) to fill the core. Then a sub-conformation generated with S_H by filling the core with H residues only. Here, we employ chain growth algorithm with random walk that places one monomer at a time by satisfying the SAW constraint. If the sub-conformation is produced successfully, we move to the Step-2. Otherwise, we restart this step (FillCore) extending the sub-sequence (maximum up to the length) that increases S_H by one at a time (at most up to the final H residue of the sequence).

Step-2 (CreatePrefix): Once the first step is successful, we create the sub-conformation for the prefix part in the reverse direction starting from $(StartPos - 1)^{th}$ position to the starting position of the sequence and then move into the next stage.

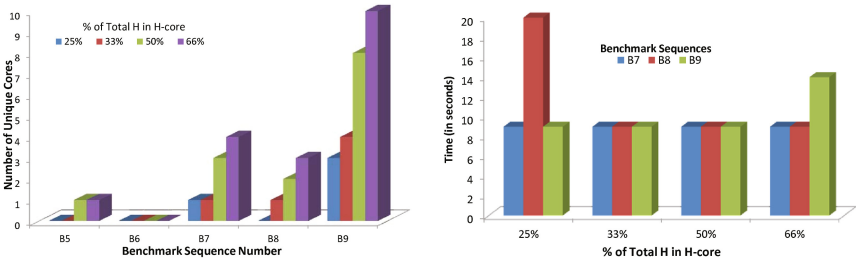
Step-3 (CreatePostfix): After the successful completion of core and prefix generation, we start creating the sub-conformation for the postfix part of the sequence starting from the immediate next residue at which the core generation ended, till the end of the sequence. It should be noted that, while generating both the prefix and postfix part, we place one move at a time satisfying the SAW constraint. However, if this step successfully generates the sub-conformation, we combine all those three sub-conformations to form a valid conformation having a core of dimension $(X_k \times Y_k \times Z_k)$. We repeat all the three steps to create I_{core} number of conformations with the particular k^{th} core dimension.

4 Experimental Results and Discussions

The performance of the proposed CIPG is evaluated with widely used benchmark sequences (defined in [4]), while the results are shown for complex sequences (i.e., B5-B9) having moderate length (i.e., 48 to 85). The comparisons are shown with the most recent technique DIG [4] as it is significantly faster than conventional random initial population generation technique. The results of the studies for the timing requirement and the quality of the initial population (number of unique cores, average and best energy) with different fractions of C_H (i.e., A_{C_H}) to generate the core for 2D and 3D HP lattice are shown in Table 1 and Fig. 2, respectively. To balance the trade-off between the time required for pre-processing and the quality of the seeds, we select 25% and 50% of C_H (i.e., $A_{C_H} = C_H/4$ and $A_{C_H} = C_H/2$) for 2D and 3D HP lattice, respectively, to generate the initial population. The sequences are executed in 25 separate runs for both the

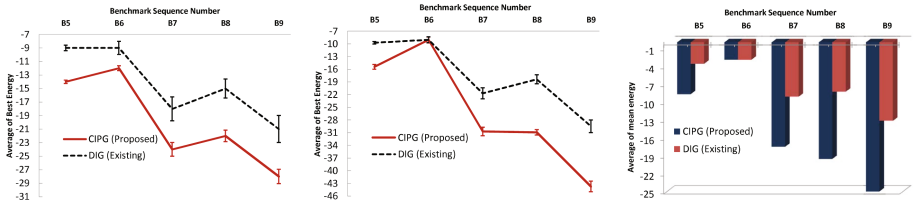
Table 1. Comparison of the proposed CIPG and existing DIG [4] for 2D HP lattice model with different A_{C_H} in terms of computation time and quality of solutions. UC* indicates number of unique cores, T# indicates required time in seconds.

	Core Dim	B5			B6			B7			B8			B9		
	(% of C_H)	Avg	UC*	T#	Avg	UC*	T#	Avg	UC*	T#	Avg	UC*	T#	Avg	UC*	T#
CIPG	50%	-6.57	8	<1	-7.11	6	<1	-13.68	15	4	-14.06	15	42	-18.51	20	46
	33%	-6.12	7	<1	-5.86	5	<1	-13.05	13	<1	-12.78	13	12	-18.37	17	15
	25%	-5.22	6	<0.5	-5.19	4	<1	-11.45	9	<1	-11.60	9	<1	-16.50	13	<1
DIG [4]		-3.48	-	<0.1	-3.65	-	<0.2	-8.06	-	<0.2	-7.93	-	<0.3	-11.65	-	<1



(a) (b)

Fig. 2. Performance analysis of proposed CIPG (a) number of generated unique cores (b) required computation time, for different percentage of Cored-H (25%, 33%, 50%, 66%) to generate initial solution



(a) (b) (c)

Fig. 3. Average of best energies with 95%CI for 5 benchmark sequences in (a) 2D lattice (b) 3D lattice. (c) Average of mean energies in 3D lattice.

proposed and existing initial population generation techniques. The average of best starting energy for the five sequences in all 25 runs, shown in Fig. 3(a), exhibits apparent superiority of the proposed CIPG over the existing DIG [4]. This superiority is even better for 3D HP lattice (Fig. 3(b)), except the sequence B6. The reason for failure in the sequence is anticipated as the small number of Cored-H as well as allowed H, that are insufficient to form any core using CIPG. The best starting energies for proposed and existing initial population generation techniques are shown in Fig. 3(b), which clearly shows that, other than $B6$, proposed CIPG starts with better energies than DIG. The 95% confidence intervals (95CI) are also calculated for all 5 sequences using both the initialization techniques, which are shown as error bars in Fig. 3(a) and Fig. 3(b), respec-

Table 2. Best energies reported by memetic algorithms initialized with proposed (CIPG) and existing (DIG) initial population generation technique with and without applying the diversification technique. E^* indicates the corresponding best energies.

	MA(without Diversification)				**MA(with Diversification)**			
	CIPG		DIG [4]		CIPG		DIG [4]	
BN(E^*)	Best	Avg±STD	Best	Avg±STD	Best	Avg±STD	Best	Avg±STD
B5(-31)	-29	-27.5 ±1.27	-27	-26.4 ±0.52	-30	-27.80 ±1.82	-27	-26.20 ±0.79
B6(-32)	-25	-24.3 ±0.82	-26	-24.4 ±0.70	-27	-23.93 ±1.49	-28	-24.73 ±1.91
B7(-54)	-47	-45.1 ±1.91	-46	-44.5 ±0.97	-51	-46.27 ±2.49	-47	-44.40 ±1.68
B8(-58)	-50	-48.6 ±1.35	-47	-42.8 ±3.08	-50	-47.33 ±1.63	-46	-42.33 ±1.80
B9(-79)	-70	-66.4 ±2.59	-63	-60.6 ±1.65	-72	-65.93 ±2.71	-62	-59.33 ±2.06

tively for 2D and 3D lattice. The 95% CI bars for the proposed CIPG show that the deviation of the energies in 25 separate runs are very little, indicating the robustness of the proposed method. Finally, the average of mean energies are shown in Fig. 3(c), which indicates the start of initial population having better average energies with the proposed method than the existing one.

Further, we have investigated the effect of initial population in optimal protein structure prediction with a simple Memetic Algorithm (MA) where both CIPG and DIG have been used separately for 3D HP lattice. We have conducted two different experiments with and without applying any diversification technique for both the MAs. All the experiments are executed for 25 runs with a maximum 100000 fitness evaluations in each run. We plot the energy values obtained for each method in Table 2 and observe that, other than B6 among the the five benchmark sequences, the conformations with the proposed CIPG technique have better fitness values than the existing DIG [4] at the end of the termination criteria. The superior performance of CIPG in both the cases (MAs with and without diversification) proves the contribution of the proposed CIPG towards successful inclusion of good seeds which lead to the convergence with better energy value in the optimization process.

5 Conclusion

In this paper, a prior knowledge based initial population generation technique has been proposed for Memetic Algorithm (MA) applied for the problem of protein structure prediction using simplified 2D and 3D HP lattice models. The proposed initialization technique speeds up the optimization process by providing quality seeds as well as by maintaining diverse solutions in the initial population. Although, only good quality seeding in initial population is not sufficient to guarantee the optimal solution or faster convergence, the variety of 'distinct' individuals seeded with prior knowledge can play significant role to explore the search space more exhaustively. An extensive analysis has been performed and comparisons have been shown with an existing technique for generating initial population.

References

1. Crescenzi, P., Goldman, D., Papadimitriou, C.H., Piccolboni, A., Yannakakis, M.: On the complexity of protein folding. In: RECOMB, pp. 61–62 (1998)
2. Lau, K., Dill, K.A.: A lattice statistical mechanics model of the conformational and sequence spaces of proteins. Macromolecules 22(10), 3986–3997 (1989)
3. Nazmul, R., Chetty, M., Samudrala, R., Chalmers, D.: Protein structure prediction based on optimal hydrophobic core formation. In: IEEE Congress on Evolutionary Computation, pp. 1–9 (2012)
4. Islam, M.K., Chetty, M.: Clustered memetic algorithm with local heuristics for ab initio protein structure prediction. IEEE Trans. Evolutionary Computation 17(4), 558–576 (2013)
5. Unger, R., Moult, J.: Genetic algorithm for protein folding simulations. Journal of Molecular Biology 231(1), 75–81 (1993)
6. Islam, M. K., Chetty, M.: Novel memetic algorithm for protein structure prediction. In: Nicholson, A., Li, X. (eds.) AI 2009. LNCS, vol. 5866, pp. 412–421. Springer, Heidelberg (2009)
7. Shymgelska, A., Hoons, H.H.: An ant colony optimization algorithm for the 2d and 3d hydrophobic polar protein folding problem. BMC Bioinformatics 6(30), 1–22 (2005)
8. Băutu, A., Luchian, H.: Protein structure prediction in lattice models with particle swarm optimization. In: Dorigo, M., et al. (eds.) ANTS 2010. LNCS, vol. 6234, pp. 512–519. Springer, Heidelberg (2010)
9. Santana, R., Larranaga, P., Lozano, J.A.: Protein folding in simplified models with estimation of distribution algorithms. IEEE Transactionson on Evolutionary Computation 12(4), 418–438 (2008)
10. Yue, K., Dill, K.A.: Sequence-structure relationships in proteins and copolymers. Physical Review E 48(3), 2267–2278 (1993)
11. Beutler, T.C., Dill, K.A.: A fast conformational search strategy for finding low energy structures of model proteins. Protein Science 5(10), 2037–2043 (1996)
12. Hoque, M. T., Chetty, M., Sattar, A.: Genetic algorithm in ab initio protein structure prediction using low resolution model: A review. In: Sidhu, A.S., Dillon, T.S. (eds.) Biomedical Data and Applications. SCI, vol. 224, pp. 317–342. Springer, Heidelberg (2009)
13. Hoque, M.T., Chetty, M., Dooley, L.S.: A hybrid genetic algorithm for 2D FCC hydrophobic-hydrophilic lattice model to predict protein folding. In: Sattar, A., Kang, B.-H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 867–876. Springer, Heidelberg (2006)
14. Bonissone, P.P., Subbu, R., Eklund, N., Kiehl, R.T.: Evolutionary algorithms+domain knowledge=real-world evolutionary computation. IEEE Trans. on Evolutionary Computation 10(3), 256–280 (2006)

EEG Based Coherence Analysis for Identifying Inter Individual Differences in Language and Logic Study

Jun-Su Kang, Swathi Kavuri, and Minho Lee*

School of Electronics Engineering, Kyungpook National University,
1370 Sankyuk-Dong, Puk-Gu, Taegu 702-701, South Korea
{wkjuns,swati.kavuri,mholee}@gmail.com

Abstract. According to Multiple Intelligences (MI) theory by Howard Gardner, human's intelligence can be divided into linguistic, logical/mathematical, musical, spatial, bodily/kinesthetic, interpersonal, intrapersonal and naturalistic areas. In this paper, two groups of students with high and low MI scores in each area for linguistic and logical/mathematical intelligences were categorized and tested based on a questionnaire in an experiment. Electroencephalogram (EEG) was recorded from 16 electrodes using Emotiv EPOC for 29 kindergarten children and 9 high school students during the assessment. To investigate the neurophysiological substrates of intelligence, EEG signal based coherence analysis in alpha, beta, gamma and theta frequency bands was performed. Our findings indicate that the group with low MI score for both language and logic showed wider and distributed cognitive activations suggesting an increased effort in processing a particular task. The group with high MI score for language showed effective connectivity within left-hemisphere and low activation in the right parietal lobe. The group with high MI score for logic/mathematics showed increased frontal activation. Performance in language and logic test was further correlated with effective connectivity in the task specific areas of brain. Based on our results we conclude that a smarter brain for language and logic is associated with the limited but affective connectivity.

Keywords: Multiple intelligence, coherence analysis, cognition, EEG.

1 Introduction

Understanding human intelligence has interesting applications in psychology as well as neuroscience. Intelligence is one of the most important psychometric constructs in the study of individual differences in cognitive abilities [1]. An understanding of these individual differences can provide improved ways to compensate for one's needs. Howard Gardner proposed Multiple Intelligence (MI) theory suggesting eight basic criteria that contribute to intelligence. MI theory not only broadened the perception of understanding intelligence beyond IQ scores [2] but also opened up new vistas of research, especially in teaching strategies for children with different predispositions in eight intelligences. It is possible that a particular strategy is more likely to be highly

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 554–561, 2013.

successful with one group of students and less successful with other groups. These kinds of personalized education strategies can make children more competitive.

The inter individual differences between people with good and poor achievements (either because of high or low aptitude or due to more or less training) can be addressed by the concept of neuronal efficiency. It is assumed that higher ability in a cognitive task is associated with more efficient neuronal processing of the task. For several years, attempts have been made to correlate intelligence with EEG. These attempts have shown a high inter individual variability [3]. In general, EEG variables that are correlated with IQ and neuropsychological test performances include power or amplitude measures and network connection measures such as coherence and phase delays [4,5]. It is interesting to note that among several reported EEG studies exploring intelligence [6,7], EEG measures indicating neural interactions showed more intense relationship with intelligence than other EEG indices, such as asymmetry indices or regional power [8]. The EEG coherence typically reports a positive correlation between neural complexity and intelligence [9]. Scalp recorded EEG coherence is a large scale measure of functional interrelation between pairs of cortical regions that is often closely related to particular cognitive task, rather than general mental ability.

In this paper we investigate the neurophysiological substrates of intelligence that contributes to a difference in language and logic/mathematics abilities of primary school children and high school students. To do this, we evaluated a subject's MI intelligence score in a particular task using performance assessment or a questionnaire. We also acquired a subject's brain wave (using wireless EEG) during performance assessment. The acquired EEG data was analyzed by a coherence analysis method in various frequency bands, in which we checked top 15 connections. The objective of the present study is to investigate the group differences in EEG coherence for excellent and poor performers in a language and logic task.

Rest of the paper is organized as follows: The theoretical aspects of this research are presented in the next section. In section 3, we present the experiment environment and results of this study. Conclusion and discussion are presented in section 4.

2 Theoretical Aspects of This Research

2.1 Multiple Intelligence(MI) Theory

MI theory was introduced by Howard Gardner. He challenged the conventional notion that intelligence can be objectively measured and reduced to a single quotient or score. He proposed the existence of at least seven basic intelligences in his book namely "Frames of Mind [10]"; Later an eighth on was added [11]. His work has encouraged educators and parents to view children as equals regardless of a quotient produced from an intelligence exam or of academic areas for which they develop competence. Practitioners of MI understand that children do not fit into a single prototype.

He sought to broaden the perception of human potential beyond the confines of traditional IQ scores, seriously questioning the validity of determining an individual's intelligence through the practice of taking the person out of his or her natural environment and asking him or her to attempt isolated tasks never done before and

probably never to be done again. Thus, He suggested educators to view intelligence as the capacity for solving problems and fashioning products in context-rich and naturalistic settings [12] rather than placing the traditional importance on the ability to produce a large quotient.

Each MI's intelligence is described in [11].

2.2 Coherence Analysis for EEG Signals

Coherence analysis of EEG signals is used to quantitatively measure the linear dependency between two distant brain regions as expressed by their EEG activities. Scalp recorded EEG coherence is a large-scale measure which depicts dynamic functional interactions between electrode signals. High coherence between EEG signals recorded at different sites of the scalp hints at an increased functional interplay between the underlying neuronal networks. A coherence value of 1 is interpreted as identifying two signals, independent of their amplitudes, whereas a coherence value of 0 corresponds to unrelated patterns of electrical activity.

With the development of computational techniques, a broader application of coherence analysis as a method to monitor frequency dependent large-scale synchronization during human intact and disturbed information processing has been established. Number of studies on EEG coherence and cognitive information processing in healthy humans has exponentially increased during the past four years. The contribution of EEG coherence analysis to the investigation of cognition and in particular language and logic is demonstrated with examples in the recent EEG studies.

Mathematically, the coherence function is obtained by cross-spectral analysis. It is an essential part of EEG spectral analysis because it enables us to quantify the relationships between different EEG signals. Detailed reviews on methodical aspects of EEG coherence analysis are given in Shaw [13], Challis and Kitney [14], etc. Here, we recall the basic definitions. Let $x_i(f)$ and $x_j(f)$ be the (complex) Fourier transforms of the time series $\hat{x}_i(t)$ and $\hat{x}_j(t)$ of channel i and j, respectively. Then the cross-spectrum is defined as

$$S_{ij}(f) \equiv \langle x_i(f) x_j^*(f) \rangle \tag{1}$$

where * means complex conjugation and < > means expectation value. Coherence is now defined as the absolute value of normalized cross-spectrum:

$$C_{ij}(f) = \frac{S_{ij}(f)}{(S_{ii}(f) S_{jj}(f))^{1/2}} \tag{2}$$

3 Experimental Setup

3.1 Classifying the Excellent Group and the Poor Group in Each MI's Area

29 kindergarten children and 9 high school students participated in the experiment. Kindergarten children were given a performance assessment related to their MI. During the performance assessment, an expert observer recorded participants' reply and

behavior. Their EEG signal was also collected at the same time. Participants' response was analyzed by Multiple Intelligence Institute in Korea, which provides each subject's MI's score. This MI's score was used to group them excellent and poor subjects. We analyzed representative brain wave of two groups with excellent MI score and worst MI score. On the other hand high school students (around 18 years old), were requested to do two surveys before the actual experiment. These two surveys include MI survey for high school students and Job aptitude psychological survey. These survey scores were used for sorting excellent and poorest subjects.

3.2 EEG Data Acquisition for Children

For children subjects, we used Emotiv EPOC wireless EEG equipment to acquire their EEG during natural performance assessment. As the preparation time for wire type EEG equipment needed about 20~30 minutes, we had to avoid children's tiredness during preparation time and we did it by using wireless EEG equipment. Participant's EEG was recorded from 16 electrodes according to the 10/20 system, as shown in Fig 1(a) and was digitalized at a rate of 128 Hz. The continuous EEG signal was referenced to a common mode sense (CMS) electrode and a driven right leg (DRL) electrode.

The EEG was recorded during the MI performance assessment. Fig. 1 (b) shows a scheme of the paradigm for recording EEG signal. First, to relax, participants played a simple game using EPOC about 3 minutes. After the relaxation step, participants took a performance assessment task for each MI's area. We recorded each MI's area, except bodily/Kinesthetic area. As the objective of this experiment was to study children's naturalistic response, we didn't consider the relaxation time for each MI area.

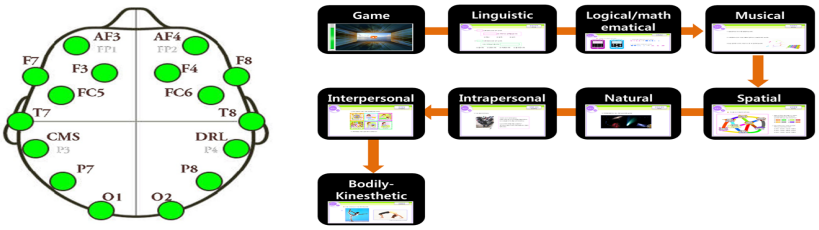


Fig. 1. (a) The green circles mark the sensor location for EEG experiment; (b) A scheme of the paradigm for recording EEG signal

3.3 EEG Data Acquisition for High School Students

Emotiv EPOC was used to record high school students EEG. The participant's EEG was collected while they took a linguistic and logical/mathematical test. We selected the test contents from metropolitan office of education's item pool. Each test included 6 problems (2 easy problems, 2 mid- level problems, 2 difficult problems). Fig. 2 shows a scheme of the paradigm for recording EEG signal.

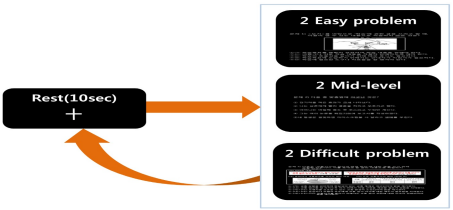


Fig. 2. Experiment scheme of the paradigm for recording EEG signal

First, participants were instructed to maintain gaze on a cross in the center of the video and start when they were ready. After 10 seconds, a test content showed up, participants were asked to avoid unnecessary body movements. At the end of each test, we conducted an off-line survey to determine the accuracy of participant's response.

4 Experimental Results

In this study, we analyzed only 2 MI areas (linguistic intelligence, logical/mathematical intelligence). Fig. 3 shows the scalp's topographical differences of a representative excellent child and poor child. In this figure, we display top 15 coherence value's location.

For the children, the excellent representative child with high MI score for language showed effective connectivity within left-hemisphere. Also the excellent representative child showed more activation on left frontal and parietal lobes and low activation in the right parietal lobe than poor representative child which means that the left hemisphere is related with language.

On the other hand, excellent child with a high MI score for logic/mathematics showed increased frontal activation which is frequently associated with arithmetic calculations in fMRI imaging studies.

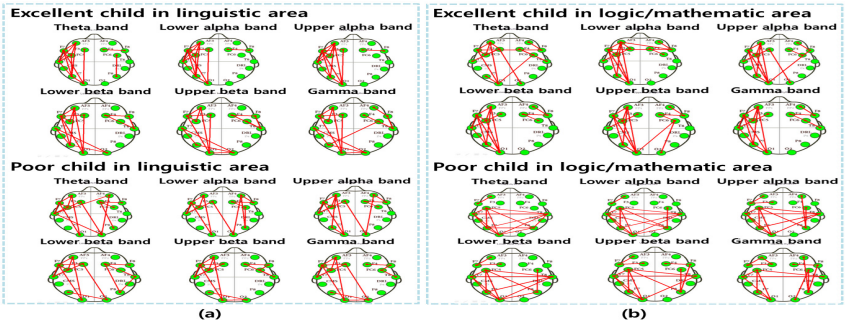


Fig. 3. EEG signal based coherence analysis in frequency bands for children (a) In the linguistic intelligence, excellent child shows weak activation on right parietal lobe in lower alpha, upper alpha and lower beta band; (b) In the logical/mathematical intelligence, excellent child activate mainly on the left cortex but poor child shows left/right cortex long connection

Fig. 4 shows differences between excellent and poor high school students. Outstanding students in linguistic intelligence showed more activation on left parietal lobe than poor students but with wider and distributed connectivity. However, for logical/mathematical intelligence, excellent students showed increased frontal activation and effective connectivity in the left frontal hemisphere that is known to be associated with arithmetic processing as compared to the poor students.

We found some interesting common patterns by comparing children's data with high school students' data. The excellent participants showed less activation on right parietal lobe than poor participants in the linguistic intelligence. Also, the excellent ones, in logical/mathematical intelligence, showed less connection between left and right cortex than the poor ones. The poor child with low MI score for both language and logic tasks showed wider and distributed cognitive activations suggesting an increased effort in processing different tasks. Performance in the language and logic test can further be correlated with effective connectivity in the task specific areas of brain. As the occipital lobe is related with visual stimuli, all subjects showed high activation on the occipital lobe.

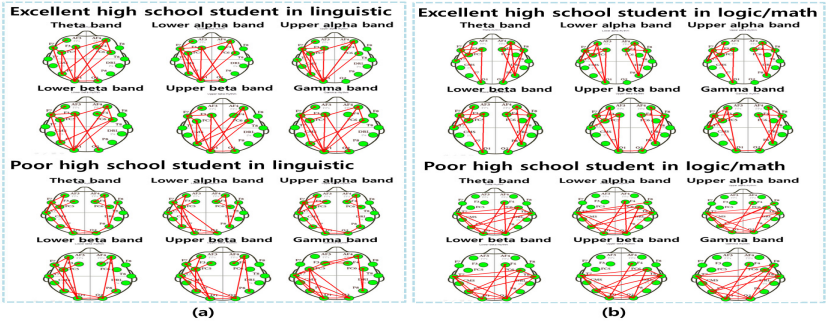


Fig. 4. EEG signal based coherence analysis in frequency bands for high school students (a) In the linguistic intelligence, excellent high school student got high activation in right parietal lobe than in left parietal lobe, (b) In the logical/mathematical intelligence, excellent high school students had weak connection between left and right cortex

5 Conclusion and Discussion

In this study, we compared excellent and poor groups of children and high school students based on their MI scores. We found effective connectivity within left-hemisphere in a language task where the excellent subjects showed less activation in the right parietal lobe than the poor ones. It has been shown that the right parietal lobe is related with visuo-spatial tasks [15] whereas left parietal lobe is mainly related with symbolic numerical information and language understanding [16]. Our results showed that the excellent group mainly uses the brain areas (left parietal lobe) corresponding to the respective cognitive tasks. For logic/mathematic intelligence, the excellent subject showed less connection between left and right hemisphere for both child and high school subjects. We argue that since the human brain is prone to constant

change, the primary brain area changes for processing language. During infancy, an excellent child mainly uses left hemisphere but as he/she grows, language, with the increase in knowledge and experiences, languages gets processed in both hemispheres. But this does not happen with poor ones and as a result they show similar connections during infancy and high school. Based on our results, we conclude that a smarter brain for language and logic is associated with limited but effective connectivity.

In our future work, we plan to analyze remaining areas of MI intelligence such as spatial, musical, natural, interpersonal and intrapersonal intelligence using various statistical methods.

Acknowledgment . This work was supported by the R&D program of the Korea Ministry of Knowledge and Economy (MKE) and the Korea Evaluation Institute of Industrial Technology (KEIT) [10041826, Development of emotional features sensing, diagnostics and distribution s/w platform for measurement of multiple intelligence from young children]

References

1. Lee, T.-W., Wu, Y.-T., Yu, Y.W.-Y., Wu, H.-C., Chen, T.-J.: A smarter brain is associated with stronger neural interaction in healthy young females: A resting EEG coherence study. Intelligence 40(1), 38–48 (2012)
2. Moran, S., Kornhaber, M., Gardner, H.: Orchestrating Multiple Intelligences. Educ. Leadership. 64(1), 22–27 (2006)
3. Giannitrapani, D.: The electrophysiology of intellectual functions. Kargere Press, New York (1985)
4. Doppelmayr, M., et al.: Intelligence related differences in EEG-bandpower. Neurosci. Lett. 381(3), 309–313 (2005)
5. Thatcher, R.W., North, D.M., Biver, C.J.: Intelligence and EEG phase reset: A two compartmental model of phase shift and lock. NeuroImage 42(4), 1639–1653 (2008)
6. Jin, S.-H., et al.: Differences in brain information transmission between gifted and normal children during scientific hypothesis generation. Brain and Cognition 62(3), 191–197 (2006)
7. Fink, A., Graif, B., Neubauer, A.C.: Brain correlates underlying creative thinking: EEG alpha activity in professional vs. novice dancers. NeuroImage 46(3), 854–862 (2009)
8. Thatcher, R.W., North, D., Biver, C.: EEG and intelligence: Relations between EEG coherence, EEG phase delay and power. Clin. Neurophysiol. 116(9), 2129–2141 (2005)
9. Ten Caat, M., et al.: High-density EEG coherence analysis using functional units applied to mental fatigue. J. Neurosci. Meth. 171(2), 271–278 (2008)
10. Gardner, H.: Frames of mind: The theory of multiple intelligences. Basic Books, New York (1985)
11. Checkley, K.: The first seven and the eighth: A conversation with Howard Gardner. Educ. Leadership. 55, 8–13 (1997)
12. Armstrong, T.: Multiple intelligences in the classroom. Association for Supervision and Curriculum Development (2009)

13. Shaw, J.C.: Correlation and coherence analysis of the EEG: A selective tutorial review. Int. J. Psychophysiol. 1(3), 255–266 (1984)
14. Challis, R.E., Kitney, R.I.: Biomedical signal processing (in four parts). Part 3: The power spectrum and coherence function. Med. Biol. Eng. Comput. 29(3), 225–241 (1991)
15. Bueti, D., Walsh, V.: The parietal cortex and the representation of time, space, number and other magnitudes. Philos. T. R. Soc. B. 364(1525), 1831–1840 (2009)
16. Tettamanti, M., et al.: Listening to action-related sentences activates fronto-parietal motor circuits. J. Cognitive Neurosci. 17(2), 273–281 (2005)

A Study on the Feasibility of Using EEG Signals for Authentication Purpose

Tien Pham[1], Wanli Ma[1,2], Dat Tran[1], Phuoc Nguyen[1], and Dinh Phung[1]

[1] Faculty of Education, Science, Technology and Mathematics,
University of Canberra, Australia
[2] Department of Computer Science, University of Houston Downtown, USA
{tien.pham,wanli.ma,dat.tran,phuoc.nguyen,
dinh.phung}@canberra.edu.au

Abstract. Authentication is to verify if one is who he/she claims. It plays an important role in security systems. In this paper, we study the feasibility of using Electroencephalography (EEG) brain signals for authentication purpose. In a general sense, there are three types of authentications including password based, token based, and biometric based. Each of them has its own merit and drawback. Technology advancing makes it possible to easily obtain EEG signals. The evidences show that finding repeatable and stable brainwave patterns in EEG data is feasible. The prospect of using EEG signals for authentication is promising. An EEG based authentication system has the combined advantages of both password based and biometric based authentication systems, yet without their drawbacks. Therefore, it makes an EEG signal based authentication suitable for especially high security system. Through the analysis and processing of EEG signals of motor imagery from BCI Competition, our experiment results confirm the theories stated in this paper.

Keywords: EEG, machine learning, pattern recognition, authentication, security.

1 Introduction

Authentication is the fundamental function of a security system to verify if one is who he/she claims. In general, there are three means of authentications: password based (something the individual knows), token based (something the individual possesses), and biometric based authentication (something the individual is). Each of them has its own vulnerabilities [3]. A password can be guessed or stolen by an imposter. Similarly, a token can be duplicated or stolen. Biometric information, such as voice, face, iris, retina, and finger print, can be recorded or photographed. Recently, EEG (Electroencephalography) signal emerges as a potential biometric modality with advantages of difficult (close to impossible) to fake, impossible to observe or intercept, unique, un-intrusive, and alive [7]. An EEG signal is a measurement of the electrical field which is generated when neurons are activated. There are a lot of studies on EEG based biometric recognition including person identification and person verification by

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 562–569, 2013.

applying machine learning [2, 6, 8, 11-13, 15-17, 18, 21]. The purpose of study so far is mainly on person recognition [5]. Although person recognition is related to authentication, the focus is different. An authentication system requires accuracy and stability, minimum risk of being faked or information disclosure, nonintrusive, easy to implement and operate, and having different credentials for different levels of security. In addition, the same person may want to set different levels of "EEG password" to the systems of different levels of security. EEG can have all those characteristics. The experimental results of these studies are very high at true positive rate that means the existing of repeatable and stable brainwave patterns correspond to mental tasks is potential. Inspired by these results, in this paper we study the feasibility of an EEG based authentication system.

The rest of the paper is organized as follows. We first introduce EEG and the application of machine learning in processing signals in Section 2. Section 3 presents using EEG for person recognition. In Section 4 and Section 5, we highlight the drawbacks of conventional types of authentication, and study the feasible to use EEG for authentication. Experiments and results are presented in Section 6. We conclude the paper with a discussion and our future work in Section 7.

2 EEG and Its Applications

An EEG signal is a measurement of the electrical field which is generated when neurons are activated [19]. EEG signals contain the information about brain activities, and they are usually recorded by placing electrodes on the scalp of a person. In this paper, we concern only human EEG.

EEG signals are divided into five major bands, delta (0.5-3 Hz), theta (4-7 Hz), alpha (8-13 Hz), beta (14-30 Hz), and gamma (>30 Hz). Delta waves are mainly associated with deep sleep and may also be observed in a waking state while theta waves are associated with creative inspiration and deep meditation. Alpha waves are the most popular in the brain activities. They appear in both relaxed awareness without attention and with concentration. Beta waves are the usual waking rhythms in the brain associated with active thinking, active attention, or solving problems. Gamma waves usually have low amplitudes, rare occurrence, and relate to left index finger, right toes, and tongue movement [19].

EEG signals have been playing an important role in health and medical applications. Epileptic seizure detection is one of the most well-known applications. Another common usage of EEG signal in health is the study of sleep disorders. In additional, the relations between EEG signals and brain diseases have been investigated [19].

Recording EEG signals is non-invasive with a portable device; therefore, EEG is widely used in Brain Computer Interface (BCI) which can provide a link between the human subject and the computer without physical contact [19]. Affective computing is also an attractive area with many researchers trying to understand the states of human minds, emotion etc. [20]. In addition, EEG is used to reveal concealed information for forensics, for example, lie detection [4].

In recent years, researchers start to establish the fact that brain-wave patterns are unique to every individual, and thus EEG signals can be used in biometrics [8].

3 Using EEG for Person Recognition

Machine learning is a branch of computer science, artificial intelligent, and pattern recognition. It has been found wide application in processing EEG signals particular in BCI [19] and also EEG-based person identification. An EEG-based person recognition system usually has two phases training and testing. There are two main components in the each of the phases feature selection and classification. First of all, the biometric information of a user is acquired from his or her brainwave signals. Next, in the training phase, EEG data pre-processing is conducted to reduce noise. Afterwards, features are extracted by a feature selection algorithm to select only useful features. Finally, these extracted feature vectors are used to train a classifier to build a model of the EEG signal patterns of the person. In the testing phase, the same feature extraction algorithm processes EEG signals of a person, and those features are compared to the models created during the training phase.

A variety of models have been used for person recognition purpose. Linear support vector machine with cross validation was employed for the classification in [2]. Neural network with spectral features was used in [16]. In [8], a Gaussian mixture model with maximum a posteriori adaptation was applied for person verification. AR (Auto Regression) coefficients with PCA (Principle Component Analysis) were used in [15]. Fisher's Linear Discriminant (FLD) was first deployed in [21] to reduce the dimensions of AR and power spectrum density (PSD) feature vectors, and then a k-Nearest Neighbours (kNN) classifier was applied.

Some more studies in applying machine learning algorithms for EEG based recognition can be found in [6, 11, 12]. Multi-sphere Support vector data description (MSSVDD) is used in [11]. MSSVDD is also used with universal background model (UBM) in [12]. In [6], J.Hu tried to analysis EEG signals for person authentication based on an ARMA (Auto-Regressive and Moving Average) model.

All this research work is concentrating on person recognition. We will review authentication in next section again, very briefly here.

4 A Review on Authentication Systems

Authentication is the foundation of any security system, in which a person is verified to be timely who he or she claims. There are 3 means of authentication: (i) something a person knows, for example, password and PIN (personal identity number), (ii) something a person has, for example, physical keys, smart cards etc., and (iii) something a person is and does – so called biometric authentication, such as voice recognition, fingerprints matching, and iris scanning etc. [9].

Authentication by something a person knows, also known as password based authentication, is the most popular authentic mechanism, where a user has to provide not only ID but also a password [3]. The system is simple, accurate, and effective. It will continue to be the working horse for authentication for many years to come [9]. However, password based authentication is not immune from malicious attacks. The popular ones are offline dictionary attack, popular password attack, exploiting user mistakes, and exploiting multiple password use [3]. The dilemma now is that with ever increasing computer power, which can crack even longer password with shorter

time, the memorability of human brain or the length password stays the same. Therefore, a feasibility alternative is extreme desirable.

Authentication by something a person has, also known as token based authentication, is an authentic mechanism that bases on objects a user possesses such as a bank card, a memory card, a smart card, and a USB Dongle [3]. This kind of authentication requires users always bringing and providing tokens when accessing the system. Presenting a token, which is not a part of a human body, can cause inconvenient. Another inconvenient of token based authentication is that all the tokens require special reader. In additional, tokens can be physically stolen, be duplicated, as well as be hacked by engineering techniques [3]. Securing the tokens is itself a challenge.

Authentication by something a person is and does, also known as biometric based authentication, tries to authenticate users based on their biometric characteristics. User characteristics can be divided into two classes including physiological characteristics such as fingerprint, face recognition, print, hand geometry, and iris recognition, and behavioral characteristics, such as hand writing, and voice etc. [3, 18]. Although biometric authentication can avoid some disadvantages of password based and token based authentication, the conventional biometrics modalities have some security disadvantages. Face, fingerprint, and iris information can be photographed. Voice could be recorded, and hand writing may be mimicked [5, 10]. Moreover, individuals can be lost or changed their biometric characteristics such as finger or face. These disadvantages require a better biometric modality for security systems.

5 Using EEG for Authentication Purpose

While the conventional types of authentication have their own shortcomings as above, EEG emerges as a potential modality for authentication because of following advantages, yet without shortcomings of the conventional types:

1. EEG is confidential because it corresponds to a mental task;
2. It is very difficult to mimic because similar mental tasks are person dependent;
3. It is almost impossible to steal because the brain activity is sensitive to the stress and the mood of the person, an aggressor cannot force the person to reproduce his or her mental pass-phrase [8]; and
4. EEG exists in every person and requires the alive person recording [1].

Therefore, we propose an authentication system using EEG signals. The system can be regarded as authentication by something a person thinks. An EEG based authentication system has two phases: enrollment and verification. In the enrollment phase, a person is asked to do some tasks, for example imagining moving a hand, a foot, a finger or the tongue, and EEG signals are recorded. For authentication purpose, which is different from person recognition purpose, the imagery tasks themselves are also a part of the credential and could not be seen by a third party. The number of tasks can be flexible and depends on the security of the system. After collecting the data, the EEG signals of each task corresponding to the user are pre-processed, extracted features, and are used to train and build the model which is kept securely in a database.

In the verification phase, when a user wants to access the system, he or she has to provide EEG signals by repeating the tasks which he/she did in the enrolment phase. These input EEG data are processed the same as in the enrolment. The obtained vector features are then fed into the classifier as testing data to match the model of the individual who he or she claims to be.

The number of tasks need to repeat depends on the requirement of the security policy. It can be just a single match or multi-match policy. A single match means the user can perform a few tasks while the verification is done by, and at least one task is matched. On the other hand, multi-match requires the EEG patterns of more than one tasks are matched for the person to be authenticated.

To use EEG for authentication purpose, EEG patterns of an individual have to be repeatable and stable on the one hand, and distinguishable from an individual to another. In this section, we study the feasibility of using EEG for authentication purpose, i.e., if EEG signals indeed have aforementioned characteristics. From the research that has been done so far on person recognition, stable patterns are feasible with motor imagination, and motor imagination is repeatable [12-14].

Therefore, we can confident that EEG patterns are stable enough for person recognition. However, more is needed to use EEG for authentication purpose than just simple person recognition. EEG based authentication system uses EEG based "password" that is the combination of personal characteristics of EEG patterns and secret task performed as seen in Table 2. The analogy can be seen in the voice based authentication system where a person can be identified from the voice, but only specific voice phrases can be used as "password", e.g., saying a special sequence of words.

6 Experiments and Results

6.1 Data Set

The Graz dataset B in the BCI Competition 2008 comes from the Department of Medical Informatics, Institute of Biomedical Engineering, Graz University of Technology for motor imagery classification problem in BCI Competition 2008 [7]. The Graz B 2008 dataset consists of EEG data from 9 subjects. The subjects were right-handed, had normal or corrected-to-normal vision and were paid for participating in the experiments. The subjects participated in two sessions contain training data without feedback (screening), and three sessions were recorded with feedback. It consisted of two classes: the motor imagery (MI) of left hand and right hand. Three bipolar recordings (C3, Cz, and C4) were recorded at sampling frequency of 250 Hz.

6.2 Feature Extraction

The signals from electrodes C3, C4, Cz were selected to extract features. The autoregressive (AR) linear parameters and power spectral density (PSD) components from these signals are extracted as features. In details, the power spectral density (PSD) in the band 8-30 Hz was estimated. The Welch's averaged modified periodogram

method was used for spectral estimation. Hamming window was 1 second 50% overlap. 12 power components in the frequency band 8-30 Hz were extracted.

Besides PSD features, autoregressive (AR) model parameters were extracted. In AR model, each sample is considered linearly related with a number of its previous samples. The AR model has the advantage of low complexity and has been used for person identification and authentication [16] [17]. Burg's lattice-based method was used with the AR model order 21, as a previous study [16] suggested when there were many subjects and epochs. The resulting feature set consists of 3*(12+21) = 99 features.

6.3 Results

The Support Vector Data Description (SVDD) method was used to train person EEG models. Experiments were conducted using 5-fold cross validation training and the best parameters found were used to train models on the whole training set and test on a separate test set. The RBF kernel function $K(x - x') = e^{-\gamma||x-x'||^2}$ was used. The parameters for SVDD training are γ and ν. The parameter γ was searched in $\{2^k: k = 2l + 1, l = -8, -7, \ldots, 2\}$. The parameter ν was searched in $\{0.001, 0.01, 0.1\}$. The best parameters found are $\nu = 0.1$, $\gamma = 2^{-3}$ for left and right hand motor imagery for each subject.

Table 1. Dataset description

Dataset	#subjects	#tasks	#trials	#sessions	Length(secs)
Graz 2008 B	9	2	120	5	7.5

Table 2. Recognition rates of 9 subjects B01-B09 using the left and right motor imagery tasks

Task / Subject	Left hand (L)	Right hand (R)	(L ∨ R)	(L ∧ R)
B01	95.3%	95.3%	99.8%	90.8%
B02	95.0%	96.3%	99.8%	91.5%
B03	96.9%	98.9%	99.9%	95.8%
B04	90.9%	92.6%	99.3%	84.2%
B05	83.0%	93.8%	98.9%	77.9%
B06	91.4%	95.9%	99.6%	87.7%
B07	94.7%	93.1%	99.6%	88.2%
B08	92.5%	93.2%	99.5%	86.3%
B09	92.6%	99.8%	99.9%	92.4%

Due to the levels of security system, tasks matched can be any of a few, e.g. ($T_1 \vee T_2 \vee T_3$), or all of them in the right order, e.g. ($T_1 \wedge T_2 \wedge T_3$). Let T_1 and T_2 be the events of correct classification of Task 1 and Task 2, respectively, we will have the probability of successfully classifying at least one of the tasks as $P(T_1 \vee T_2) = P(T_1) + P(T_2) - P(T_1 \vee T_2)$.

Two designated motor imagery tasks were to cue left hand and right hand as seen in Table 1. If the system is more important, all sequence tasks must be matched, so with 2 tasks T_1 and T_2 the probability of successful access will be $P(T_1 \vee T_2) = P(T_1) * P(T_2)$. Table 2 summarizes the two figures $P(L \vee R)$ and $P(L \wedge R)$ for 9 subjects for authentication with tasks.

Table 2 shows that the probability of successful access is very high when the single match policy is applied with only two tasks ($L \vee R$). It is also seen that security is considerable strengthened when the authentication system applies the multiple match policy ($L \wedge R$).

7 Discussion and Future Work

Using EEG signals for authentication has the advantages of both password based and biometric based authentications, yet without their drawbacks. Firstly, EEG signals are biometric information of individuals, and have the advantages of biometric based authentication, yet EEG based authentication can overcome the disadvantages of conventional biometric based authentication.

On the other hand, EEG based authentication brain patterns correspond to particular mental tasks, and they are considered as passwords. As the result, EEG based authentication has all the benefits of password based authentication, yet without the vulnerabilities.

Moreover, security systems may have a multiple security levels with EEG based authentication because it can be adjusted by the number of matched tasks. If a system is of a lower security level, an individual can perform a few tasks, and the system only requires that at least one task is matched. If a system is of a high security level, all tasks in the sequence also in the right order must be matched, so it helps to strength the security system.

Using EEG for authentication purpose is feasible, and also desirable. In the near future, we will investigate the EEG based authentication on a large dataset. The impact of different individuals performing the same task and the same single person performing different tasks will also be studied. Repeatable and stable EEG patterns also remind us a research direction in which EEG based biometric is combined with cryptography effectively for information security applications, and we will study the possibility of using the task sequence as the key for encryption.

References

1. Allison, B.: Trends in BCI research: progress today, backlash tomorrow? The ACM Magazine for Students 18, 18–22 (2011)
2. Ashby, C., Bhatia, A., Tenore, F., Vogelstein, J.: Low-cost electroencephalogram (EEG) based authentication. In: 2011 5th International IEEE/EMBS Conference on Neural Engineering (NER), pp. 442–445 (2011)
3. Brown, L.: Computer Security: Principles and Practice. William Stallings (2008)

4. Grubin, C., Madsen, L.: Lie detection and the polygraph: A historical review. The Journal of Forensic Psychiatry & Psychology 16, 357–369 (2005)
5. He, C., Chen, H., Wang, Z.: Hashing the MAR Coefficients From EEG Data For Person Authentication. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2009, pp. 1445–1448 (2009)
6. Hu, J.: Biometric System based on EEG Signals by feature combination. In: 2010 International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), pp. 752–755 (2010)
7. Leeb, R., Brunner, C., Muller-Putz, G., Schlogl, A., Pfurtscheller, G.: BCI Competition 2008 - Graz data set B, http://www.bbci.de/competition/iv/
8. Marcel, S., Millán, J.R.: Person authentication using brainwaves (EEG) and maximum a posteriori model adaptation. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(2007), 743–752 (2007)
9. Ma, W., Campell, J., Tran, D., Kleeman, D.: Password Entropy and Password Quality. In: 2010 4th International Conference on Network and System Security (NSS), pp. 583–587 (2010)
10. Matyáš, V., Řiha, Z.: Security of biometric authentication systems. In: 2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM), pp. 18–28 (2010)
11. Nguyen, P., Tran, D., Le, T., Hoang, T.: Multi-sphere support vector data description for brain-computer interface. In: 2012 Fourth International Conference on Communications and Electronics (ICCE), pp. 318–321 (2012)
12. Nguyen, P., Tran, D., Le, T., Huang, X., Ma, W.: EEG-Based Person Verification Using Multi-Sphere SVDD and UBM. In: 17th Pacific-Asia Conference, pp. 289–300 (2013)
13. Nguyen, P., Tran, D., Huang, X., Sharma, D.: A Proposed Feature Extraction Method for EEG-based Person Identification. In: The International Conference on Artificial Intelligence (ICAI 2012), USA (2012)
14. Nguyen, P., Tran, D., Huang, X., Ma, W.: Motor Imagery EEG based Person Verification. In: Rojas, I., Joya, G., Cabestany, J. (eds.) IWANN 2013, Part II. LNCS, vol. 7903, pp. 430–438. Springer, Heidelberg (2013)
15. Palaniappan, R.: Two-stage biometric authentication method using thought activity brain waves. International Journal of Neural Systems 18 (2008)
16. Poulos, M., Rangoussi, M., Alexandris, N.: Neural network based person identification using EEG features. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP 1999, pp. 1117–1120 (1999)
17. Poulos, M., Rangoussi, M., Alexandris, N., Evangelou, A.: Person identification from the EEG using nonlinear signal classification. Methods of Information in Medicine 41(1), 64–75 (2002)
18. Rathgeb, C., Uhl, A.: A survey on biometric cryptosystems and cancelable biometrics. EURASIP Journal on Information Security (2011)
19. Sanei, S., Chambers, J.: EEG signal processing. Wiley-Interscience (2007)
20. Schaaff, K., Schult, S.: Towards emotion recognition from lectroencephalographic signals. In: 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops, ACII 2009, pp. 1–6 (2009)
21. Yazdani, A., Roodaki, A., Rezatofighi, S.H., Misaghian, K., Setarehdan, S.K.: Fisher linear discriminant based person identification using visual evoked potentials. In: 9th International Conference on Signal Processing, ICSP 2008, pp. 1677–1680 (2008)

Learning to Detect Frame Synchronization

Yingying Wang, Chun Zhang, Qi Peng, and Zhihua Wang

Institute of Microelectronics,
Tsinghua University, Beijing 100084, China
{yywangqhd,qipeng08}@gmail.com,
{zhangchun,zhihua}@tsinghua.edu.cn

Abstract. Conventional frame synchronization detection algorithms in the communication system, such as correlation and maximum likelihood, either fails to deal with frequency deviation problem or requires great efforts to be implemented on the hardware platform. Therefore, we parallel the frame synchronization detection problem with the Multi-instance Learning (MIL) in the field of machine learning, and propose a novel Learning to Detect Frame Synchronization algorithm (LDFS). This algorithm is mainly conducted offline to learn preamble signal detectors, which can then be efficiently implemented on the hardware platform to accomplish the synchronous detection task. In this paper, we first solve frame synchronization detection problem from the point of machine learning, and thus our algorithm displays some advantages over the existing algorithms. First, the resulted detector is simple and efficiently realized on the hardware platform. Second, the learned detector is adaptive to work under different communication frequencies straightforwardly without extra modifications. Experimental results demonstrate the effectiveness and promise of the proposed LDFS algorithm.

Keywords: Frame synchronization detection, Multi-instance Learning, Signal Processing.

1 Introduction

In communication system, such as RFID and WiFi, data is transmitted as the format of frame, which consists of the message information following synchronization marker (preamble) embedded in the bitstream. The preamble signal indicates the beginning of message and the accurate transmission frequency [1]. Detecting frame synchronization is composed of a series of tasks that: recognizing the preamble signal, labeling its location, adjusting the sampling frequency, and preparing for the reception of subsequent message information. Since the accuracy of the frame synchronization detection determines the success of communication establishment, the detector of frame synchronization shall be reliable and robust against the interference signal and the variation of communication frequency in a practical communication system.

Conventional frame synchronization detection methods can be classified into two groups, which are correlation-based methods and maximum likelihood methods. Since the preamble signal is often defined as the local autocorrelation function

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 570–578, 2013.

with sharp unimodal characteristics, the correlation methods [2][3] were proposed to detect the preamble signal indicated by the correlation peak, through correlating the local preamble sequence with the received sequence sequentially. These correlation techniques, which often obtain satisfactory performances in dealing with high Signal-Noise-Ratio (SNR) signals, are easily understood and realized on hardware platform. However, they cannot be applied in non-ideal communication system, due to their sensitives to the variation of signal frequency. In order to solve the frequency deviation problem, [4] introduced an improved correlation algorithm based on the differential demodulation decisions, allowing for less than 8% frequency offset, but this anti-offset property will be damaged seriously when the offset grows gradually.

Another kind of frame synchronization detection method, i.e. maximum likelihood, which has an improved performance in fading channel, was proposed by [5] to maximize the tolerance of frequency offset. However, this algorithm requires great efforts to be realized on the hardware platform and huge expenses in detecting frame synchronous online. [6][7] proposed the Likelihood Ratio Test (LRT) and Generalized likelihood Ratio Test(GLRT) algorithms, which make tradeoffs between the detection accuracy and cost of hardware realization. However, they have less-than-ideal performances with low SNR signal as input.

Different from the existing frame synchronization detection algorithms, we propose to detect the preamble signal in a novel approach by paralleling the frame synchronization detection problem with the Multi-instance Learning (MIL) in the field of machine learning. The problem of Multi-instance Learning is first studied by Dietterich [8][9] for drug activity prediction. In that problem, each drug molecule is seen as a bag, which is composed of different instances, corresponding to diverse low-energy shapes of this molecule. It is difficult to know which shape of the molecule is active, but it is explicit to distinguish whether the molecule is active or not. Thus multi-instance learning is proposed as a weakly supervised learning paradigm, and receives a great deal of attention. A variety of improved MIL algorithms have been developed over the past years. In particular, Andrews et al. [10] formulated the multi-instance learning problem with support vector machines and proposed mi-SVM algorithm, that relabels the instances in positive bags using the learned decision hyperplane; Zhang et al. [11] focused on the most positive or least negative instance for a positive or negative bag, and presented the EMDD method.

The goal of frame synchronization detection is to distinguish the preamble signal out from the other kinds of signal in a basic communication system. We classify the preamble signal and other signals into two different categories, then the frame synchronization detection can be interpreted as a classification task, which is a classical problem in the field of machine learning. Moreover, in a practical communication system, we are not provided with the particular communication frequency while conducting frame synchronization detection, and thus we cannot figure out the true length of the preamble signal sequence. However, if we assume that there is a specified range of the allowed communication frequency, then the length of the true preamble signal would also be in a certain range, whose upper bound is determined

by the lowest allowed communication frequency; likewise, the lower bound is determined by the highest allowed communication frequency. Therefore, the preamble signal candidate must appear as a signal sequence fraction corresponding to the aforementioned longest length, as shown in Fig. 1. If we regard this longest signal sequence as a bag, then all its fraction signal sequences of the lengths larger than the lower bound can be seen as the instance and have probabilities to be the preamble signal. Following this idea, we regard the frame synchronization detection as a multi-instance learning problem, and propose the Learning to Detect Frame Synchronization algorithm (LDFS). This algorithm has two advantages: 1) the preamble detector can be efficiently learned offline, and then effectively used for online detecting; 2) since the detector is learned through comprehensively considering the frequency deviation, its performance under different communication frequencies can be well ensured. Experimental results demonstrate the effectiveness and promise of the proposed LDFS algorithm.

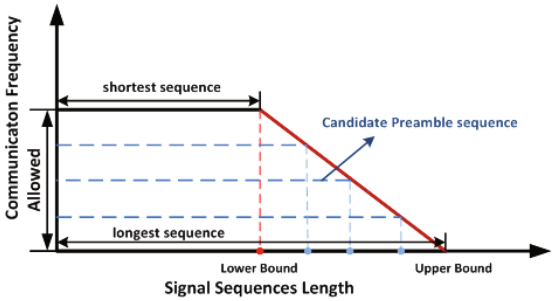


Fig. 1. Illustration of the possible preamble signal sequence

2 Problem Formulation

We implement the LDFS in the RFID communication system based on $IEC18000-6C$ protocol, where the preamble signal is composed of 6 symbols, as shown in Fig. 2. In practice, signal would be interfered by the noise when propagating through physical communication medium, which degrades the signal-noise ratio. Since there is no reference clock in a RFID Tag to calibrate the communication frequency, the transmitted signal from RFID tag may have a large frequency deviation up to $\pm$ 22%, which increases the difficulty in frame synchronization detection of RFID reader.

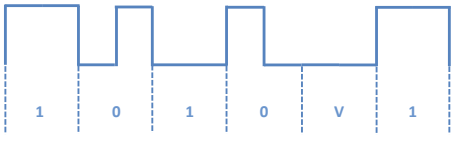


Fig. 2. Preamble signal sequence composed of 6 symbols in RFID protocol

If the sampling frequency we use to transform the analog signal into digital signal is 8 times of communication frequency, the ideal preamble signal of 6 symbols can be formulated to a 48-dimensional vector x. However, if the sampling frequency is fixed, while exists communication frequency deviation, the length of generated preamble vector would not be constant ever. Given $\pm$ 22% frequency deviation, the length of preamble signal would vary from 40 to 62, corresponding to the +22% and −22% communication frequency deviation respectively. Therefore, the preamble signal candidate can appear as a signal sequence fraction corresponding to the longest length 62. If we regard this longest signal sequence as a bag, then all its signal sequences fraction of the lengths larger than 40 can be seen as the instance and have probabilities to be the preamble signal. Following this idea, we regard the frame synchronization detection as the multi-instance learning problem, and propose the Learning to Detect Frame Synchronization algorithm (LDFS). The flowchart of learning to detect frame synchronization consists of the offline training and online detecting, as shown in Fig.3

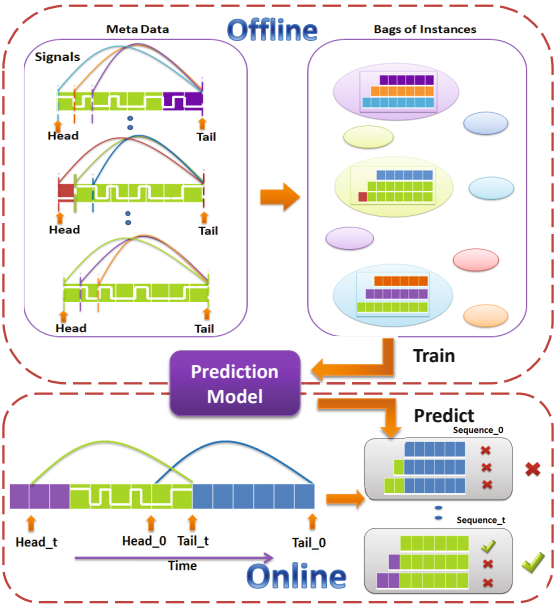


Fig. 3. Flowchart of learning to detect frame synchronization

In particular, we construct a queue with a fixed length 62 to save the signal sequence, and generate the bags and their instances. At a particular time, the new coming signal would be inserted into the tail of the queue, and the oldest signal would be popped from the head of the queue at the same time. We can regard the queue as a bag, then the vectors extract from the queue with different start indexes but with the same end index at the tail of the queue can be considered as its instances. If there is a preamble signal in the queue (bag), it must be represented by one of its extracted vectors (instances). After a period of time, we

can obtain a serious of queues (bags) $Q = \{X_1, \cdots, X_n\}$, where n is the number of bags. As for the queue (bag) X_i. All its extracted vectors (instances) make up the set $\{x_{i1}, \cdots, x_{im}\}$, where m is the number of instances in the current bag. Assume that instance label $y_{ij} \in \{-1, +1\}$ indicates whether the instance x_{ij} in the bag X_i represents the preamble signal or not. If a bag X_i contains a preamble signal, there shall be one positive instance label and then we assign the bag label $Y_i = 1$ to this bag. Likewise, if a bag X_i has no preamble signal, all its instances label are negative, and then we assign the bag label $Y_i = -1$. Formally this constrait can be written as:

$$\sum_j \frac{y_{ij}+1}{2} \geq 1, \forall i \ \texttt{s.t.}\ Y_i = 1 \quad \texttt{and}\ \ y_{ij} = -1,\ \forall i \ \texttt{s.t.}\ Y_i = -1 \tag{1}$$

For simplicity, we consider a linear function to predict whether the label for the instance x_{ij} is positive or negative,

$$y_{ij} = sign(f_{ij}), \quad f_{ij} = wx_{ij} + b. \tag{2}$$

Since instances in a particular bag are represented by vectors of different lengths, we have to learn a group of weight vectors $\{w_1, \cdots, w_m\}$. In the large margin principle, similar with SVM, the final objective function can be formulated as:

$$\begin{aligned} \min_{\{y_{ij}\}} \min_{w,b,\xi} \ & \frac{1}{2}\sum_{j=1}^{m} \|w_j\|^2 + C\sum_{i,j}\xi_{ij} \\ \texttt{s.t.}\ \ & y_{ij}(w_j x_{ij} + b_j) \geq 1 - \xi_{ij},\ \xi_{ij} > 0,\ y_{ij} \in \{-1, 1\} \\ & \sum_j^m \frac{y_{ij}+1}{2} \geq 1, \forall i \ \texttt{s.t.}\ Y_i = 1 \quad \texttt{and}\ \ y_{ij} = -1,\ \forall i \ \texttt{s.t.}\ Y_i = -1 \end{aligned} \tag{3}$$

where C is a constant selected via cross validation.

3 Optimization

An alternating optimization procedure can be utilized to solve the problem (3) by iteratively updating $\{y_{ij}\}$ and w_j until convergence, which includes the following two steps: 1) For given instance labels $\{y_{ij}\}$, solve the QP program for each w_j and find the optimal discriminant functions; 2) For given discriminant functions, update one, several or all instance labels $\{y_{ij}\}$ in a way that locally minimize the objective. Note that we re-initialize the QP-solver at every iteration with the previously found solution, which will usually result in a significant speed up. In terms of initialization of optimization procedure, we suggest to impute positive labels for instances in positive bag as the initial configuration. The complete iteration optimization procedure is illustrated in Algorithm 1.

After the optimization process, we can obtain the objective detection model composed of m weight vectors $\{w_1, \cdots, w_m\}$ and biases $\{b_1, \cdots, b_m\}$, which will be implemented in the hardware architecture. In order to balance the detection speed and accuracy, we extract instance vectors of lengths varying from 40 to 62 at the step of 2, and thus we have $m = 12$.

Algorithm 1. Algorithm for LDFS

Input: $\{x_{ij}, y_{ij}\}$ **for** $i \subset Q$, $j = 1, 2...m.$, ϵ_0, err
while $err > \epsilon_0$ **do**
  for $j = 1, ...m$ **do**
    $\{\mathbf{w}_j, b_j\} \leftarrow$ QP Solution $\{x_{ij}, y_{ij}\}$ **for** $i \subset I$, $j = 1, 2...m.$
    $f_{ij} = \langle \mathbf{w_j}, x_{ij} \rangle + b_j$, **for** $i \subset I$, $j = 1, 2...m.$
  end for
  for $i \subset I$ **do**
    if $Y_i = -1$ **then**
      $y_{ij} = -1$
    end if
    if $Y_i = 1$ **then**
      $y_{ij^\star} = sign(f_{ij})$ `s.t.` $f_{ij^\star} = \max_j(f_{ij})$, $j = 1, ..m$
    end if
  end for
  $err = \frac{1}{2}\sum_{j=1}^{m} \|w_j\|^2 + C\sum_{i,j} \xi_{ij}$
end while
Output: $\{\mathbf{w}_j, b_j\}$, $j = 1, 2..m$

For online detection, we use Verilog to programm the frame synchronization detection module, which is composed of an offline learned detection model, a shifter to save the received signal sequence, a Finite State Mechine (FSM) as the control center and an interrupt response unit for output.

4 Experiment

In this section, we evaluate the effectiveness of the proposed LDFS algorithm and compare it with the correlation and maximum likelihood (ML) algorithms under different simulations of communication environment.

4.1 Experimental Setup

We compare the preamble detection performance of different algorithms by considering two important factors in communication system, i.e., signal-noise ratio (SNR) and frequency deviation. Under different conditions of SNR and frequency deviation, we use a queue of length 62 to record the received signal sequences by RFID reader. The dataset can be constructed by copying the data of queues (bags), starting from the time that the preamble signal sequence enters the queue and stopping when the preamble signal sequence exits the queue. Specifically, by fixing the communication frequency as 240KHz, we change the SNR from 0 dB to 12 dB at the step of 1 to collect the data under different SNR conditions. Likewise, by fixing the SNR as 10 dB, we consider ±24% deviation of the 240 KHz communication frequency at the step of 4% to collect the data under different frequency deviation conditions.

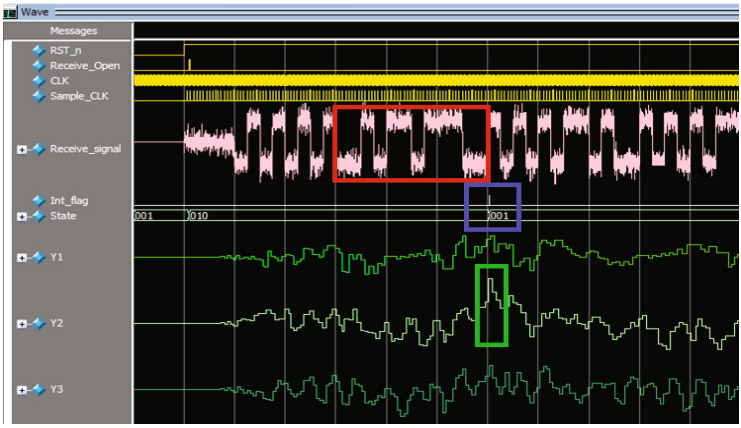


Fig. 4. Simulation waves on ModelSim

We split the dataset equally to obtain the training set and test set. The offline training process is accomplished on the training set to generate the preamble detection model, which would be evaluated on the test set in the accuracy rate.

4.2 Experimental Results

We sampled a fraction of preamble signal of $10dB$ SNR and $280KHz$ frequency, whose length is 42 under the sampling frequency 8 times over 240KHz communication frequency, for simulation with ModelSim 6.5c. On Figure 4, the prediction score Y_2 of the 2-th detection function (w_2, b_2), has a peak value at the end of preamble signal sequence. This score exceeds the pre-defined threshold of frame synchronization detection, whereas the other scores $Y_i (i \neq 2)$ only have some oscillations at that time. Hence, the preamble detector module successfully recognizes the preamble signal and approximates its corresponding communication

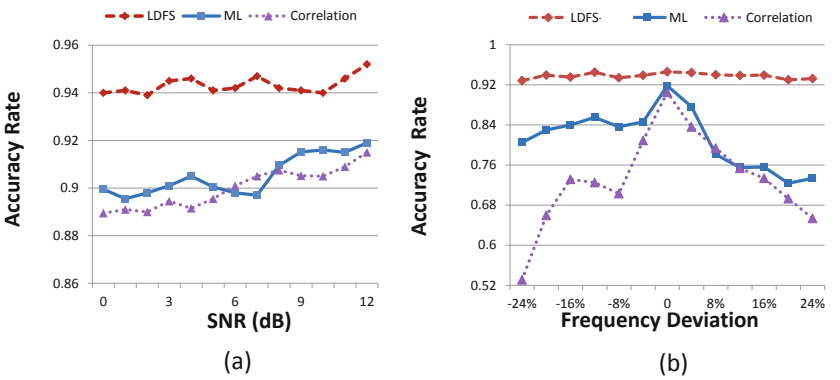


Fig. 5. Accuracy rate comparisons of different algorithms

frequency. Once the preamble signal has been detected, the interrupt response unit produces a pulse signal, the finite state mechine enters into the next state, and then the preamble signal detection task with the frequency deviation is accomplished.

We compare the LDFS to both the correlation and maximum likelihood algorithms for frame synchronization detection task on two datasets under different SNR and frequency deviation conditions. We group the test data into different subsets according to their SNRs, and then report the detection results of different algorithms under specifical SNRs in Figure 5 (a). From the result, we find that the detection accuracy rate of LDFS outperforms other competitors under different SNRs stably. As for the frequency deviation comparison, we similarly group the test data into different subsets, and the detection results are shown in Figure 5 (b). For example, LDFS obtains about 25.0% and 10.0% relative improvements in accuracy rate compared with those of the correlation and maximum likelihood algorithms at -12% frequency deviation point respectively.

5 Conclusion

In this paper, we parallel the frame synchronization detection problem with the Multi-Instance Learning (MIL) in the field of machine learning, and propose a novel Learning to Detect Frame Synchronization algorithm (LDFS), which has two advantages: 1) the preamble detector can be efficiently learned offline, and then effectively used for online detecting; 2) the learned detector is adaptive to work under different communication frequencies straightforwardly without extra modifications. Experimental results demonstrate the proposed LDFS algorithm is effective and efficient.

Acknowledgments. Thanks for Chang Xu's kind assistance. This research was partly supported by National High Technology Research and Development Program of China (863 Program). (No. 2011AA010404) and Major National S&T Program of China (No.2011ZX01034-001-001-2).

References

1. Mengali, U., D'Andrea, A.N.: Synchronization techniques for digital receivers. Springer (1997)
2. Barker, R.H.: Group synchronizing of binary digital systems. Communication Theory, 273–287 (1953)
3. Sklar, B.: Digital Communications. Prentice-Hall, Englewood Cliffs (1988)
4. Scholtz, R.: Frame synchronization techniques. IEEE Transactions on Communications 28(8), 1204–1213 (1980)
5. Massey, J.: Optimum frame synchronization. IEEE Transactions on Communications 20(2), 115–119 (1972)
6. Chiani, M., Martini, M.G.: Optimum synchronization of frames with unknown, variable lengths on Gaussian channels. In: IEEE Global Telecommunications Conference, GLOBECOM 2004, vol. 6, pp. 4087–4091. IEEE (2004)

7. Chiani, M., Martini, M.G.: Practical frame synchronization for data with unknown distribution on AWGN channels. IEEE Communications Letters 9(5), 456–458 (2005)
8. Foulds, J., Frank, E.: A review of multi-instance learning assumptions. Knowledge Engineering Review 25(1), 1 (2010)
9. Zhou, Z.H., Zhang, M.L., Huang, S.J., et al.: Multi-instance multi-label learning. Artificial Intelligence 176(1), 2291–2320 (2012)
10. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: Advances in Neural Information Processing Systems, pp. 561–568 (2002)
11. Zhang, Q., Goldman, S.A., Yu, W., et al.: Content-based image retrieval using multiple-instance learning. In: Machine Learning-International Workshop then Conference, pp. 682–689 (2002)

Reinforced Explorit on Optimizing Vehicle Powertrains

Victor Parque, Masakazu Kobayashi, and Masatake Higashi

Toyota Technological Institute,
2-12-1 Hisakata, Tempaku-ku, Nagoya 468-8511, Japan
{vparque,kobayashi,higashi}@toyota-ti.ac.jp

Abstract. How to build optimal vehicular powertrains? We study this question and propose an algorithm inspired by a domain-general design process. The basic idea is to interplay co-biasingly between the local approximations of discrete design and the global refinements of continuous parameters. The proposed method was evaluated to design powertrains of four types of vehicles: Series Hybrid Electric Vehicle(SHEV), Parallel Hybrid Electric Vehicle(PHEV), Fuel Cell(FC) and Electric Vehicle(EV). Simulation results show noticeable improvements on mileage per gas emissions over different study cases. To our knowledge, this is the first study aiming at designing vehicle powertrains considering the holistic point of view.

Keywords: Series Hybrid Electric Vehicle, Parallel Hybrid Electric Vehicle, Fuel Cell, Electric Vehicle, Reinforcement Learning, Explorit.

1 Introduction

How to build optimal vehicular design layouts? Answering this question implies developing effective search heuristics and compact knowledge representations for meaningful designing. Often, such heuristics use rules that link preferential and functional requirements and design parameters[1,2]. How to build such rules?

The general problem is to map functional requirements to concrete definitions so that the whole designing optimises pre-stated goals[3]. Similarly, effective design of vehicular layouts means optimising the powertrain structure and the design parameters such that the relation of fuel consumption to gas emissions is maximised. During the last decade, researchers have mainly focused on parameter optimization of pre-defined vehicle powertrains[4,5,6,7,8,9,10]. However, the holistic view on how to build optimal vehicular systems has remained elusive.

Here we introduce a simple and effective algorithm for problems involving optimal structure and parameter configuration. The basic idea is to optimise the structure and parameters as interdependent and co-biasing problems: *Sarsa-Learning* is used to map functional definitions to concrete hardware realizations, and *Explorit* is used to optimise the parameter configurations. The aim of combining these heuristics is to map the fast approximations of online learning(in discrete space) to the global refinements of parameter optimization(in continuous space), so that the overall optimisation is focused on improving the most promising regions in the vehicular search space. Our approach aims at contributing towards the holistic design of things.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 579–586, 2013.

2 Reinforced Explorit

Reinforced Explorit(RE) uses the concept of co-search by linking the learning of structural discrete approximations with the searching of parameter definitions for optimal design layouts. It also uses the general representation ability of graphs for sequential decision making problems. The phenotype structure is inspired by evolutionary graphs[11,12], where state machines are allowed to evolve; but the learning algorithm extends the interplay between exploration and exploitation as a search mechanism. This section describes the structure and the algorithm.

2.1 Structure

An individual in *Reinforced Explorit* models a *process P* as a network of connected *node functions* $P = \{F_1, ..., i, j, ..., F_N\}$, where the node i comprises both a set of discrete *means definitions* $M(i) = \{i_1, i_2, ..., i_I\}$ and a set of continuous parameters $p(i) = \{p_1, p_2, ..., p_{M_i}\}$, where M_i is the number of continuous parameters of node i. The search space is a vector with elements of the set $P \bigcup p(i), \forall i \in P$. By representing a process P as a network wit sub-nodes and parameters, we aim to model a machine as a concatenation of singular processes, where each node represents a functional component, and each connection represents the interaction and flow among components. Fig. 1 shows an example of two functional components (node i and j) and their flow interaction. The connectivity(network topology) and the number of functional components(vertices) is known a priori. We assume a ring topology(the node transition is fixed), and leave the problem of building variable networks for future work.

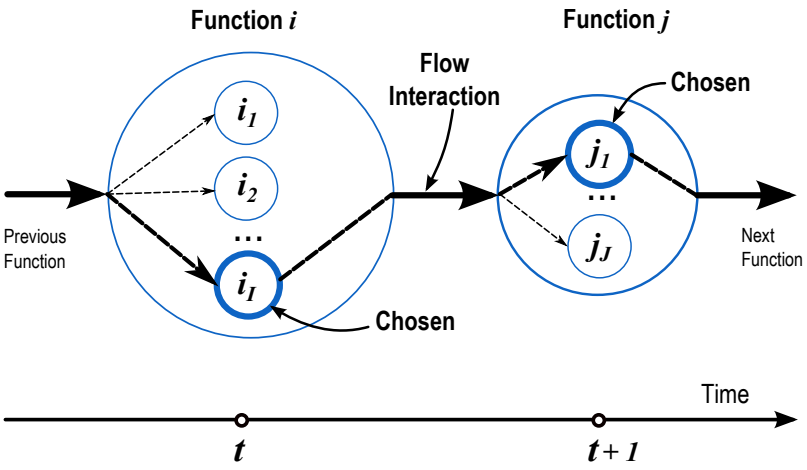


Fig. 1. Basic phenotype and node transition of an individual in *Reinforced Explorit*

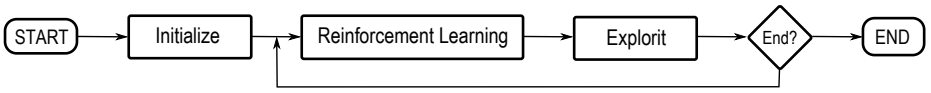


Fig. 2. Flowchart of Reinforced Explorit

2.2 Optimisation Algorithm

Fig. 2 shows the general algorithm of *Reinforced Explorit*. The basic idea is to map and link the local approximations of desirable solutions to the performance of global refinements iteratively. That is,

- *Q-learning* has the role to approximate the mappings of every function node i in the process P to its means $M(i)$ so that $Q-values$ represent the degree of desirability to link the function i to the means $M(i)$; and
- *Explorit* has the role of refining these approximations through systematic exploration and exploitation of the space concerned with the process P.

During the ***Reinforcement Learning***, *Q-values* are updated following:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\Big[r_t + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\Big], \tag{1}$$

where *state* s refers to *node function* i, t refers to the ordinal index in the node transitions, *action* a_t refers to the *means* $M(i)$ chosen for *node function* i, α refers to learning rate, γ refers to discount rate, and r_t refers to the reward obtained by choosing the *action* a_t at state s_t during time t. Rewards are computed following:

$$r_t = tanh\Big[\frac{f(P')}{f(P)} - 1\Big], \tag{2}$$

where $f()$ means "*fitness function of*", and P' refers to the state of the process P after taking *action* a_t at *state* s_t. The reason of using the above to compute rewards during the learning phase is to quantify the fitness improvements of the process P after changing the discrete *means* definitions for each *node function* $i \in P$. The function $f()$ is problem dependant.

Fig. 1 shows an example of the node transition to update the Q-values in Sarsa:

1. At time t, the node i refers to the set $M(i)$, and selects one element using ϵ-greedy policy: a maximum Q-value among $Q(i,i_1), Q(i,i_2), \ldots, Q(i,i_I)$ with probability $1-\epsilon$, or a random one with probability ϵ. Let us assume Q_{i_l} is selected.
2. Set i to i_I, observe reward r_t. The next node becomes node j.
3. At time $t+1$, the node j selects an element from $M(j)$ following the same procedure of step 1. Let us assume Q_{j_1} is selected.
4. Q-values are updated as follows: $Q(i,i_I) \leftarrow Q(i,i_I) + \alpha\Big[r_t + \gamma Q(j,j_1) - Q(i,i_I)\Big]$
5. $t \leftarrow t+1$, $i \leftarrow j$, $i_I \leftarrow j_1$. Go to step 2.

The advantage of using Reinforcement Learning to map node functions to means definitions is economical: the increase of the number of node functions increases the number of states linearly; whereas in the conventional *Q-learning* case, *state* would

refer to total information from the environment, and *action* would refer to current available option to the agent.

During the ***Explorit*** phase, the space $S = p(i), \forall i \in P$ is optimised keeping P constant. We use *Explorit*, due to its feature to avoid premature convergence by interplaying exploration and exploitation systematically[13]. The basic idea behind *Reinforced Explorit* is to model an organism to maintain the stable energy incomes while searching *entities* E with high degrees of value, quality and novelty in a *space* S.

The space S is a collection of *entities* E with cardinality $|S| = \prod_{i=1}^{D} \eta^E$, where D is the dimensionality of S, and η^E is the number of discrete divisions for all dimensions of S. While novelty is computed using set theory, value and quality are computed using quantiles q_R over improvements of the objective function. To guide its search, the organism uses a generalized heuristic and adaptive memory elements. Improvements found by *Explorit* are used to update the *Q-values* for the Reinforcement Learning phase, where the improvements are computed as follows:

$$r_E = tanh\Big[\frac{f_E}{f_{\bar{E}}} - 1\Big], \tag{3}$$

where $f_{\bar{E}}$ and f_E refer to the fitness function of process P before and after *Explorit* is executed. Since Q-values approximate the degree of *desirability* to link the function requirements $i \in P$ to their means specifications $M(i)$, the above has the role of explicitly biasing the initial approximations of the Q-values to obtain more accurate definitions of such *desirability*. Thus, the basic search behaviour is co-dependant, i.e., *Reinforced Explorit* uses online learning to approximate the desirability of solutions, and continuous optimization to improve such approximations globally.

3 Simulations

3.1 Simulation Conditions

Leaving adaptive approaches in parameter tuning for future work, we use Sarsa with ϵ, α and γ set to 0.1, 0.1, and 0.9 respectively. As for Explorit, we set η^E and q_R to 50 and 0.75, respectively. Furthermore, in the best of our knowledge, there is no previous benchmark on optimizing vehicle layouts on a discrete-continuous approach, so we use a well known derivative-free algorithm(Genetic Algorithm of the Global Optimization Toolbox in Matlab) as the main benchmark. For both algorithms, the maximum number of evaluations is set at 500.

3.2 Vehicle Models

Advisor[1] was used as the vehicle simulation tool[14]. ***Advisor*** enables the modelling transient vehicle behaviour by using measured data through look-up tables, rather than equations, thus making simulations reasonably accurate. Fig. 3 shows the types of vehicle powertrains used, and the Table 1 shows the list of vehicle components and variables.

[1] `http://bigladdersoftware.com/advisor/`

The objective function is designed to maximise fuel economy and minimise gas emissions following:

$$f = \frac{MPGGE}{1 + HC + CO + NOX}, \tag{4}$$

where $MPGGE$ refers to the consumption of miles per gallon gasoline equivalent(mi/gal), and HC, CO and NOX refer to hydrocarbon, carbon monoxide and oxide of nitrogen(g/mi). These quantities are computed over two different driving cycles: FTP-75 cycle(urban) and HWFET cycle(highway) as shown by Fig. 4. Furthermore, in order to ensure quality in driving requirements, the vehicles are evaluated with additional testings, as shown by the list of constraints in Table 2.

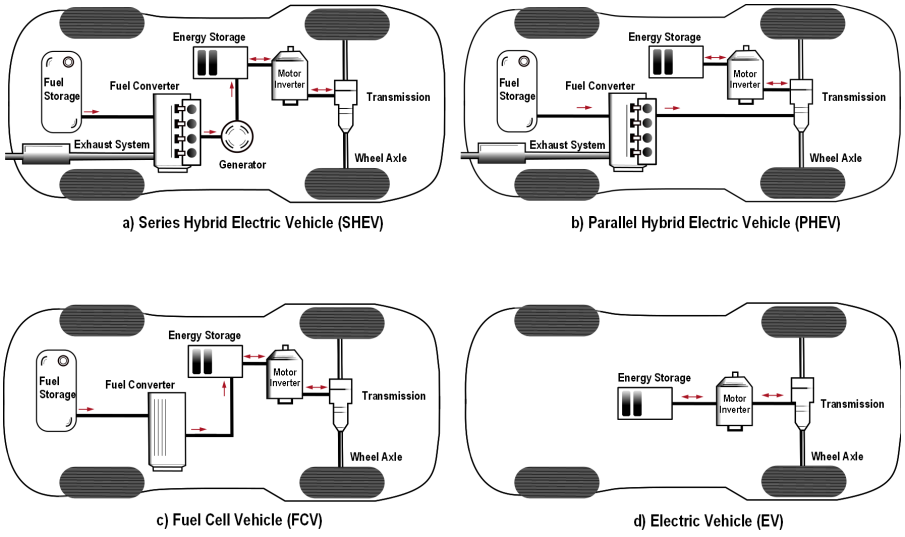


Fig. 3. Vehicle Models Used in Advisor

3.3 Simulation Results

In order to evaluate the quality and behaviour of fitness convergence, Fig. 5 shows the fitness values versus the number of fitness evaluations, where the performance over 25 independent runs shows the average, lower and upper bounds of the fitness performance.

Note that Reinforced Explorit(RE) is an individual based algorithm, compared to the population based approach of the referential algorithm(GA). Thus the first line plots of GA follow after evaluating the whole population in the first generation. In both cases, RE and GA stop after 500 fitness evaluations.

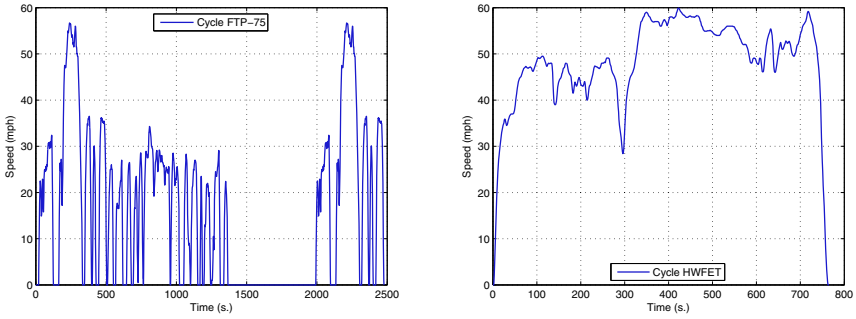
Fig. 5 shows that the convergence speed of the proposed method is comparable or superior to the referential algorithm, where the proposed method outperforms consistently in the cases of Series Hybrid Electric Vehicle and the Parallel Hybrid Electric Vehicle. Furthermore, these results show that Reinforced Explorit avoids getting trapped in premature convergence, as it occurs with the referential heuristic in the *Series* powertrain.

Table 1. Vehicle components and parameters for each vehicle in ***Advisor***

SYMBOL	ADVISOR VARIABLE		VEHICLE TYPE			
	Name	Description	SHEV	PHEV	FC	EV
F_1	'energy_storage'	Battery system	✓	✓	✓	✓
F_2	'motor_controller'	Motor and Inverter system	✓	✓	✓	✓
F_3	'fuel_converter'	Fuel converter system	✓	✓	✓	×
F_3	'generator'	Generator system	✓	×	×	×
F_4	'transmission'	Transmission system	✓	✓	✓	✓
F_5	'exhaust_aftertreat'	Exhaust and catalyst system	✓	✓	✓	×
F_6	'wheel_axle'	Wheel and axle system	✓	✓	✓	✓
p_1	'ess_module_num'	Number of modules in the battery.	✓	✓	✓	✓
p_2	'ess_cap_scale'	Scaling capacity for battery.	✓	✓	✓	✓
p_3	'fc_pwr_scale'	Scaling factor for power.	✓	✓	✓	×
p_4	'gb_spd_scale'	Speed parameter for gearbox.	✓	✓	✓	✓
p_5	'gb_trq_scale'	Torque for gearbox.	✓	✓	✓	✓
p_6	'mc_spd_scale'	Speed factor for motor.	✓	✓	✓	✓
p_7	'mc_trq_scale'	Torque factor for motor.	✓	✓	✓	✓

Table 2. Performance constraints for each vehicle in ***Advisor***

Constraint	Description
Gradeability Test	At 88.5 km/h for 20 min. at Curb Weight and 5 passengers and cargo(408kg) = 6.5%
Acceleration Test	Time for 0-96.5km/h ≤11.2 s. Time for 64-96.5km/h ≤ 4.4 s. Time for 0-137km/h ≤ 20 s.
Drive Cycle	Difference between drive cycle requested speed and achieved speed at every second during the drive cycle ≤ 3.2 km/h
State of Charge	(Final battery state - Initial battery state) ≤ 0.5%



(a) FTP-75 (b) HWFET

Fig. 4. Drive cycles used in simulation includes (a)city and (b)highway

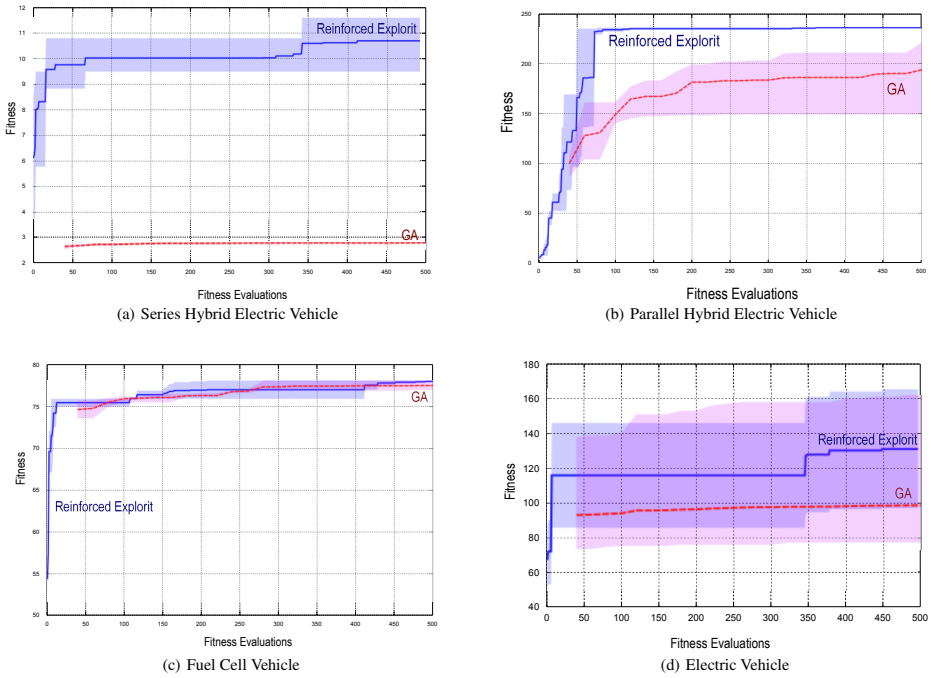


Fig. 5. Fitness values during the first five hundred evaluations in the used vehicle powertrains.

The reason of this behaviour is mainly due to the fact of interplaying between exploration and exploitation systematically and co-biasingly, where Sarsa Learning roughly approximates the desirable solutions in the discrete space and Explorit aims at refining such approximations through global updates in the Q-values.

4 Conclusion

Inspired by a domain-general design process, we have proposed *Reinforced Explorit* as an optimization algorithm for discrete-continuous space. The main idea is to link the initial approximations of online learning to the enhanced global refinements in the search space. Simulation using relevant vehicles show that the proposed method obtains vehicle powertrains with improved ratio of mileage to gas emissions. Our future work aims at building driveable modular vehicles. Furthermore, we will study the applicability of *Reinforced Explorit* on solving mixed integer nonlinear problems in manufacturing and engineering problems.

References

1. Karniel, A., Reich, Y.: Multi-level modelling and simulation of new product development processes. Journal of Engineering Design 24(3), 185–210 (2013)
2. Suh, N.P.: The Principles of Design. Oxford University Press, New York (1990)

3. Wang, L., Amos, H.C., Deb, K.: Multi-objective Evolutionary Optimisation for Product Design and Manufacturing. SpringerLink (2011)
4. Montazeri-Gh, M., Poursamad, A., Ghalichi, B.: Application of genetic algorithm for optimization of control strategy in parallel hybrid electric vehicles. Journal of the Frankling Institute 343, 420–435 (2006)
5. Balesdent, M., Bĕrend, N., Dĕpincĕ P., Chriette, A.: A survey of multidisciplinary design optimization methods in launch vehicle design. Structural and Multidisciplinary Optimization 45, 619–649 (2012)
6. Wu, L., Wang, Y., Yuan, X., Chen, Z.: Multiobjective optimization of HEV fuel economy and emissions using the self adaptive differential evolutionary algorithm. IEEE Transactions on Vehicular Technology 60(6), 2458–2470 (2011)
7. Wu, J., Zhang, C.H., Cui, N.X.: PSO algorithm-based parameter optimization for HEV powertrain and its control strategy. International Journal of Automotive Technology 19(1), 53–69 (2008)
8. Mi, C., Masrur, A., Gao, D.W.: Hybrid Electric Vehicles Principles and Applications with Practical Perspectives. John Wiley & Sons (2011)
9. Wang, L., Collins, E.G., Li, H.: Optimal design and real-time control for energy management in electric vehicles. IEEE Transactions on Vehicular Technology 60(4), 1419–1429 (2011)
10. Chan, C., Bouscayrol, A., Chen, K.: Electric, Hybrid, and Fuel-Cell Vehicles: Architectures and Modeling. IEEE Transactions on Vehicular Technology 59(2), 589–598 (2010)
11. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial Intelligence through Simulated Evolution. John Wiley & Sons (1966)
12. Mabu, S., Hirasawa, K., Hu, J.: A Graph-Based Evolutionary Algorithm: Genetic Network Programming (GNP) and its Extension Using Reinforcement Learning. Evolutionary Computation 15(3), 369–398 (2007)
13. Parque, V., Kobayashi, M., Higashi, M.: Explorit for Global Optimization. In: NIPS Workshop on Optimization for Machine Learning, Lake Tahoe, Nevada, USA (2012)
14. Markel, T., et al.: ADVISOR: A systems analysis tool for advanced vehicle modeling. Journal of Power Sources 110, 255–266 (2002)

Decoding and Predicting Implicit Agreeing/Disagreeing Intention Based on Electroencephalography (EEG)

Suh-Yeon Dong, Bo-Kyeong Kim, and Soo-Young Lee

Department of Electrical Engineering,
Korea Advanced Institute of Science and Techonology (KAIST),
Daejeon, 305-701, Republic of Korea
suhyeon.dong@gmail.com,
{kbghome,sylee}@kaist.ac.kr

Abstract. A new experiment design is proposed to understand human implicit intention by using electroencephalography (EEG). EEG data is recorded using 32-channel electrodes while seeing various sentences which contain self-relevant contents. Subjects are asked to make a decision of agreement or disagreement just after sentence ending is shown. Based on their answer, support vector machine is used for pattern classification with radial basis function kernel. The classification result shows the intention to the sentences can be classified with 67.89% of maximum average accuracy. The spatial relationship of average classification accuracy shows right frontal areas have relatively high classification accuracy. Our findings indicate that covert representation of agreement or disagreement intention can be found in the EEG band power and it is also possible to predict subjects implicit intention even before making explicit expression.

Keywords: Implicit intention, EEG band power, support vector machine.

1 Introduction

Understanding others intention is an important aspect in communications between human. Human mainly interacts by speech with others, but also explicit expressions like facial expression and gestures can be used to emphasize. However, explicit information may not be enough to understand what human really intends to. Furthermore, whether it is intended or by accident, humans explicit expression is not always the same with his true intention. If machine learned only interpreting one or two explicit expressions, it may work well only some fixed command in a limited environment, but not in the natural situation. Thus, understanding human implicit intention is necessary to improve current human computer interacting system. Here we define implicit intention as a hidden intention not explicitly expressed, but reflect his real mind at this moment.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 587–594, 2013.

Most well-known application of implicit intention study is a lie-detector. When telling a lie, deceptive intention is explicitly expressed while a goal is figuring out implicitly reflected truth. As many lie-detection researches have been done with brain activities [1–3], brain signals cannot be intentionally deceived or hidden. Thus, we assume that neural activities can be used to measure implicit intention. Modern techniques enable to measure brain activities associated with thoughts, feelings, and behaviors. Scalp recorded electroencephalography (EEG) has a significant advantage of non-invasiveness and high temporal resolution in recording brain signals. Event-related potential (ERP) is the most frequent used analysis techniques to observe neural activities in time domain. Motor intent can also be detected by ERP observations, before actual movement or even during imagination of moving [4, 5]. However, our goal is to understand implicit intention which is a higher level of cognition. Even though human does not express his mind, machine can detect what he wants now and do so. In this study, it is investigated that either agreement or disagreement towards given statements on the basis of personal experience as two types of intention. We assumed brain activity, especially at the frontal sites, precedes the conscious decision for how to answer, so neural activities in pre-decision stage can help us to predict how they are going to answer in real decision stage. Many studies have been trying to investigate the frontal EEG dynamics in working memory task [6], and theta activities in frontal EEG in mental activity [7]. However there is no previous research of decoding implicit intention in a higher cognitive task.

2 Methods

2.1 Subjects

Nine healthy right-handed Korean subjects (6 males and 3 females) were recruited from the students community. They are undergraduate or graduate students, and voluntarily participated. All participants did not have a history of psychiatric disorder, significant physical illness, head injury, neurological disorder, and alcohol or drug dependence. After complete explanation of the study, written informed consent was obtained from all subjects. The study was submitted to the regular review in the institutional review board and approved.

2.2 Materials

74 stimulus sentences were selected for the experiment. Sentences were chosen from the list of Minnesota multiphasic inventory (MMPI) which is one of the most frequently used for psychological tests. Selected sentences were identified into two types; affirmative and negative sentence which are all written in Korean. In linguistic typology, Korean is a SOV language because the subject, object, and verb of a sentence appear in this order. If English follows SOV, “I read a book” becomes “I book read”. Also, selected sentences had only one type of a verb, an existence verb that indicates simply the existence of objects in the

sentence. Allowing negations, verbs are either "to be" or "not to be", and due to the word order, verbs appear at the end of the sentences.

Sentences were asking his/her experience or opinion then after reading a complete sentence, subjects were asked to explicitly express either agreement or disagreement by responding "yes" or "no". To prevent other sentence components interferences, unnecessary adverbs or adjectives were all left out. Also, some components which can imply negative form such as "any", "at all", "even once" etc. were removed.

2.3 Experimental Paradigm

Experiment consists of two trials interleaved with break between them. Subjects were asked to see 37 sentences in each trial, so 74 sentences were shown for one subject. Type of sentences was classified according to the sentence end either positive or negative sentence, but the order of presentation was random to avoid that subjects can be aware of which type is coming next. Fig. 1 shows the experiment paradigm.

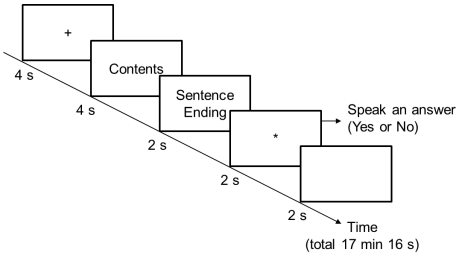


Fig. 1. Experiment design. Each sentence part is separately presented. Subjects were asked to respond "Yes" or "No" when the asterisk (*) is shown on the screen.

2.4 Data Acquisition

The EEG was recorded from BrainAmp system (Brain Products GmbH, Germany) and 32 electrodes of an EEG cap (BrainCap). 30 electrodes were placed on the scalp according to the International 10-20 system. One electrode for recording eye movement (EOG) was positioned below subjects left eye. One other electrode dedicated to the electrocardiogram (ECG) was placed on subjects collarbones in left side. The impedance of each electrode was maintained below $10k\Omega$ using gel.

2.5 Preprocessing

Data were acquired with a sampling rate of 500Hz, along with 60Hz notch filtering. As widely known in the field, raw EEG signals are highly contaminated with

various noises. There are movement artifacts made by human such as eye blink, muscle, or heartbeat as well as artifacts caused by electrical power lines. First, acquired EEG signals were high-pass filtered with a cut-off frequency at 1Hz and transition bandwidth 0.2Hz in order to remove line noise. Movement artifacts cannot be eliminated easily because one artifact affected many channels simultaneously. Therefore, independent component analysis (ICA) was widely used to find artifact-related independent component [8, 9]. ICA is a statistical method that maximizes the mutual independence of components. So ICA enables to select contaminated independent component, and reconstruct uncontaminated signals. In this study, extended ICA in EEGLAB was used to extract independent components [10], and artifact components related to the eye blinks, eye movements, and ECG were removed.

3 Analysis and Results

We focused on whether subjects agree or disagree on the statements while seeing contents block. Before sentence ending is presented, decision that agrees or disagrees on those statements may be determined in the contents block. Considering the EEG signal characteristics, averaging is the most simple and efficient techniques to diminish the artifacts and noises on the signal as well as to investigate the representative response of the experimental task. However, single trial classification has become an important approach for application-purpose in the real life, such as brain-computer interface (BCI). In this study, single trial classification in frequency domain has been done using pattern classification. Overall analysis procedure will be introduced: feature extraction, classification, and the result.

3.1 Feature Extraction

EEG oscillations have been related to a variety of functions such as perception, cognition, sleep, etc. For a long time, researchers have found the sensory and cognitive processes are modulated by synchronous neural activity which is in turn induced by oscillations. A variety of studies have demonstrated that neural oscillations like frontal midline theta are closely associated with memory processes and mental activities [6, 7]. In this study, features were extracted from the neural oscillations during the task (reading sentence contents) and applied to the nonlinear classifier. The procedure for feature extraction is illustrated in fig. 2. For each trial, the 4-s time epoch during contents block is extracted from the preprocessed signal of an EEG channel. The power spectrum is obtained by applying N-point Fast Fourier Transform (FFT). To extract features for underlying neural oscillations related to human implicit intention, 6 frequency bands are considered: delta ($2-4$ Hz), theta ($4-8$ Hz), alpha ($8-13$ Hz), beta1 ($13-20$ Hz), beta2 ($20-30$ Hz) and gamma ($30-40$ Hz). Concatenating spectral powers for 6 frequency bands leads to a feature vector of each channel. Each feature vector is normalized to become the sum of entries to be one.

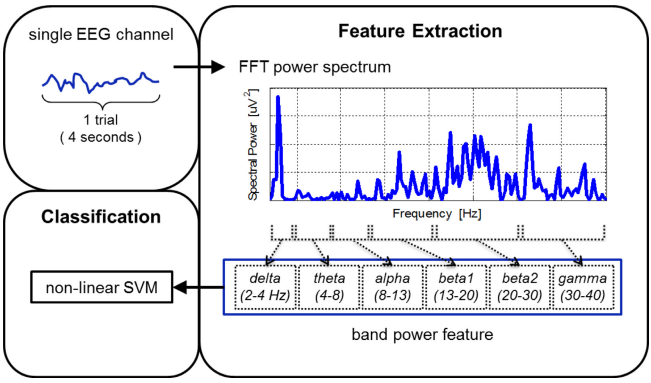


Fig. 2. Feature extraction procedure

3.2 Classification

Input feature vectors are applied into the nonlinear classifier. Classification has done into two steps: training phase and testing phase. In training, 80% of input samples are used to train classifier with known labels. After training, rest of 20% input samples are applied to the pre-trained classifier and predict the labels of testing samples. Then classification accuracy can be obtained. This procedure is repeated for 5 times changing the dataset of training and testing for reliable performance evaluation. It is called 5-fold cross-validation. Classification performance is evaluated for each channel and each subject. There are many classifier for pattern classification, we selected support vector machine (SVM), and used LIBSVM tool [11]. The radial basis function (RBF) kernel is the most popular kernel function used in SVM classification. RBF kernel on two samples x_i and x_j is defined as,

$$K(x_i, x_j) = exp(-\frac{||x_i - x_j||^2}{2\sigma^2}), \tag{1}$$

but it is possible to make it simple using parameter $\gamma = \frac{1}{2\sigma^2}$,

$$K(x_i, x_j) = exp(-\gamma||x_i - x_j||^2), \gamma > 0. \tag{2}$$

The objective function with soft margin also includes the question of selecting appropriate slack parameter C. As seen in Eq. 3, slack parameter, also called penalty parameter, C decides the contribution of ξ_i, which is the degree of misclassification, on the objective function.

$$min_{\mathbf{w},b,\xi_i} = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \tag{3}$$

It is conventional to find optimal kernel parameter γ and penalty parameter C in training phase using grid search [12]. We find the optimal γ and C which make high training accuracy and apply to the testing data.

3.3 Result

Our experimental design demonstrates and decodes implicit intention for each single trial using the selective attention algorithm. Average recognition rate during 5-fold cross validation is obtained from the subject-dependent classification at 30 EEG channels. The maximum mean accuracy is achieved at FP2 channel as 67.89%. In fig.3, right frontal areas have relatively high classification accuracy distribution. It is considered as the representative brain pattern involved in implicit intention process

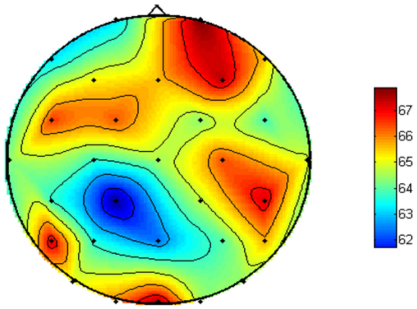


Fig. 3. Average classification accuracy distribution on the scalp illustration

In the right frontal channels, frequency band characteristics are also analyzed using Fisher's score (FS). We calculated FS for each frequency band power in 4 selected right frontal channels as following relationship in eq.4.

$$FS = \frac{N_1(\mu_1 - \mu)^2 + N_2(\mu_2 - \mu)^2}{N_1\sigma_1^2 + N_2\sigma_2^2} \tag{4}$$

In eq.4, μ_i indicates sample mean of class i, σ_i^2 is sample variance of class i, N_i is number of samples in class i, and μ is overall sample mean. Higher FS value means well-discriminative feature of two classes. For any single frequency band, FS values are distributed within 0.01 to 0.03, whereby sample data in one class are not clearly distinguishable with those in other class. However, we observe beta2 band power has relatively higher FS value compared to other frequency band powers, commonly in 4 selected right frontal channels (Fp2, F4, FC2, and Fz). On the basis of this finding, beta2 band power has major contribution of implicit intention recognition.

4 Discussion

Machine understanding human is a key objective of this field of research. Starting from the assistant robot which recognizes limited number of speech commands, the multimodal sensory approach is currently increasing. Also EEG-based BCI is rapidly growing. The common aim of various types of human computer interface is translating user behavior or bio signals into machine commands. However, there are not many higher-level cognition tasks even in the EEG-based approach, while there were a lot of simple tasks such as motor imagery. Polygraph was one of the higher task which draws attention in the field. An established system had been troubled due to a lot of malfunctioning in natural situation. In order to make lying situation, subjects were asked to tell a lie regardless of his real intention. Sometimes, they were asked to answer reversely to their thoughts. Reaction from the required action may not be able to reflect his true intention. These differences may cause a malfunctioning in the real situation. In this study, whether subject tells a truth or lie is not our concern. While decision-making related to the personal issue, his real intention may be revealed in neural activities. Thus, we predicted implicit intention states using EEG band power, particularly with high accuracy at the right frontal sites. We defined implicit intention is an intention which can be observed in neural activities, but not explicitly found yet. In decision making of agreement and disagreement, we found implicit intention is generated in a brain before making an answer explicitly.

5 Conclusion

In this study, we aimed to discriminate implicit intentions, particularly either agreement or disagreement towards given statements, based on a conventional nonlinear SVM. Our experimental design employed separate presentation of sentence contents and either a positive or negative ending. Nonlinear SVM classifies EEG band power features with higher average accuracies in the right frontal channels. Maximum recognition rate of 67.89% is achieved at FP2 channel. The spatial distribution of average classification accuracy is investigated by the topographic accuracy maps. Furthermore, using Fisher's score, high beta band power has relatively high contibution to classification. In conclusion, the proposed experimental paradigm can deduce a subjects implicit intention while reading self-relevant sentences and our pattern classification approach can decode implicit intention with high accuracy in frontal EEG channels.

Acknowledgment. This research was supported by the Brain Research Program through the National Research Foundation of Korea funded by the Ministry of Science, ICT & Future Planning (2013-035100).

References

1. Wolpe, P.R., Foster, K.R., Langleben, D.D.: Emerging neurotechnologies for lie-detection: Promises and perils. The American Journal of Bioethics 10(10), 40–48 (2010), PMID: 20945266

2. Simpson, J.R.: Functional mri lie detection: Too good to be true? Journal of the American Academy of Psychiatry and the Law Online 36(4), 491–498 (2008)
3. Abootalebi, V., Moradi, M.H., Khalilzadeh, M.A.: A new approach for EEG feature extraction in p300-based lie detection. Computer Methods and Programs in Biomedicine 94(1), 48–57 (2009)
4. Wang, T., Deng, J., He, B.: Classifying eeg-based motor imagery tasks by means of time-frequency synthesized spatial patterns. Clinical Neurophysiology 115(12), 2744–2753 (2004)
5. Soon, C.S., Brass, M., Heinze, H.-J., Haynes, J.-D.: Unconscious determinants of free decisions in the human brain. Nature Neuroscience 11(5), 543–545 (2008)
6. Onton, J., Delorme, A., Makeig, S.: Frontal midline EEG dynamics during working memory. NeuroImage 27(2), 341–356 (2005)
7. Kazutoyo, I.: Frontal midline theta rhythm and mental activity. Psychiatry and Clinical Neurosciences 52(6), 555–566 (1998)
8. Jung, T.-P., Humphries, C., Lee, T.-W., Makeig, S., McKeown, M.J., Iragui, V., Sejnowski, T.J.: Removing electroencephalographic artifacts: comparison between ICA and PCA. In: Proceedings of the 1998 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing VIII, pp. 63–72 (1998)
9. Jung, T.-P., Humphries, C., Lee, T.-W., Makeig, S., McKeown, M.J., Iragui, V., Sejnowski, T.J.: Extended ica removes artifacts from electroencephalographic recordings. In: Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10, NIPS 1997, pp. 894–900. MIT Press, Cambridge (1998)
10. Delorme, A., Makeig, S.: Eeglab: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. Journal of Neuroscience Methods 134(1), 9–21 (2004)
11. Chang, C.-C., Lin, C.-J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
12. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification. In: Department of Computer Science and Information Engineering. National Taiwan University, Taipei (2003)

A New On-Line Learning Method for Coping with Recurring Concepts: The ADACC System

Ghazal Jaber[1,2,3], Antoine Cornuéjols[1,2], and Philippe Tarroux[3]

[1] AgroParisTech, UMR 518 MIA, F-75005 Paris, France
[2] INRA, UMR 518 MIA, F-75005 Paris, France
[3] Université de Paris-Sud, LIMSI, Bâtiment 508, F-91405 Orsay Cedex, France
{ghazal.jaber,antoine.cornuejols}@agroparistech.fr,
philippe.tarroux@limsi.fr

Abstract. When the environment changes, as is increasingly the case when considering unending streams and long-life learning tasks, it is necessary to rely on on-line learning with the capability to adapt to changing conditions a.k.a. concept drifts. Previous works have focused on means to detect changes and to adapt to them. Ensemble methods relying on committees of base learners have been among the most successful approaches. In this paper, we introduce a new second-order learning mechanism that is able to detect relevant states of the environment in order to recognize recurring contexts and act pro-actively to concepts changes. Empirical comparisons with existing methods on well-known data sets show the advantage of the proposed algorithm.

Keywords: online learning, ensemble methods, concept drift.

1 Introduction

Recent years have witnessed the emergence of a whole new set of applications involving data streams made of pairs $(\mathbf{x}_t, y_t)$, where the "answer" or true label y_t is revealed (sometimes long) after the input $\mathbf{x}_t$. Additionally, it is often the case that the unknown target function evolves with time, something known as *concept drift*. These new needs have spurred a surge of research works geared towards the development of adaptive strategies [1]. Most of them operate either by passively tracking the evolving concept or by using an explicit detection mechanism of concept changes before launching an adaptation or relearning process.

However, better learning strategies may take advantage of the examination of the history of past concept in order to anticipate likely future changes or to recognize when a past concept recurs. We have developed ADACC (*Anticipative Dynamic Adaptation to Concept Change*), a system that uses this kind of second order learning to accelerate its adaptation to changing conditions in the environment. This is accomplished through an ensemble method that controls a pool of incremental learners. Our main innovation lies in the design of a method to select decisive concepts in the history of the system. This enables the recognition

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 595–604, 2013.

of recurring concepts and opens at the same time the possibility of learning the underlying trends. The latter will be described elsewhere.

In the following, noticeable existing works on online learning in the context of concept changes are described in Section 2. ADACC is presented in Section 3 while Section 4 reports our empirical results and a comparison to the state of the art methods. Section 5 concludes with possible avenues for future work.

2 Relevant Works

In presence of concept drift, the central concern it to learn from data relevant to the regularities currently governing the world. Several directions have been proposed to dynamically adapt the memory of the past data. One is to use time sliding windows with a fixed or a varying size [2]. This necessitates either a priori knowledge about the dynamics of the environment, or a good heuristic that guesses when to shrink or expand the sliding window. Another approach relies on weighting past data with respect to their relevance to the current situation [3]. Again, this is all dependent on the design of an appropriate weighting mechanism. By contrast with these types of approaches that explicitly control the memory of past data, another set of techniques relies instead on the control of past hypotheses. Thus, ensemble-based approaches maintain a pool of on-line learners that each maintain their own memory of the past and compete for giving advice on the current query $\mathbf{x}_t$. By managing the population of these base-learners, and possibly their weights, based on their current prediction performance, one is implicitly controlling the use of past data [4].

A specially interesting case is the one of recurring concepts. This may happen for instance when the environment is subject to seasonal variations which repeats over time. This problem introduces a new challenge in that apparently obsolete data or learned concepts may well be relevant again in the future. It would thus be profitable to exploit this past knowledge as soon as it is appropriate rather than learn anew from the incoming data stream. This requires, however, that interesting memory traces be stored and that their relevance to the current situation be quickly recognized. In recent years, several proposals have been put forward to meet this challenge, particularly within the ensemble-based approach (see for instance [5–9]). They either work at the level of the examples themselves, by chunking them in some way, possibly organizing a hierarchy of chunks, or they work at the level of the learned concepts, trying to learn significant concepts in the stream of examples while avoiding redundancy.

This paper presents a new technique which solves both the problem of detecting regularities (concepts) that are significant and new, and the problem of recognizing when to use past knowledge. It is supposed that the data stream is governed by piecewise stationary environments with concept shifts in between. We show how an ensemble-based method can be used to detect stable environments and how a memory of past concepts can be built which both avoids redundancies and permits a quicker recognition of relevant past concepts than a purely adaptive approach. Unlike other systems handling concept recurrence

[6, 7], our method does not recognize new concepts using a drift detection system. Thus, it avoids the difficulties inherent in having to detect gradual concept changes while still being robust to false alarms.

3 Concept Changes: Adaptive and Anticipative

The proposed technique, called ADACC, implements an anticipative mechanism, able in particular to exploit recurring concepts, build on top of an ensemble-based adaptive online learner. We first briefly describe the latter one.

3.1 Adapting to Concept Changes

The main idea is to maintain a pool of base learners $\{h_t^i\}_{1 \leq i \leq N}$, each of them adapting to the new input data, and to administer this pool or ensemble thanks to a strategy for inserting and deleting base learners. Sketchily:

- Each *base learner* in the pool continuously adapts with new incoming data until it is removed from the pool.
- Every τ time steps, the base learners are *evaluated* on a window of size τ_{eval}.
- Based on the results of this evaluation, the deletion procedure *chooses* a base learner to be removed.
- A new base learner is *created* and inserted in the pool. It is protected from possible deletion for a duration τ_{mat}.
- For each new incoming instance $\mathbf{x}_t$, the prediction $H(\mathbf{x}_t)$ results from a *combination* of the prediction of the individual base learners $h_t(\mathbf{x}_t)$.

Variations around this general framework lead to specific algorithms [4, 10–13]. For instance, after extensive testings, we converged on the following settings. The *evaluation* procedure counts the number of erroneous predictions on the last τ_{eval} time steps. The *deletion strategy* randomly selects one base learner from the worst half of the pool evaluated as above. The *global prediction* uses the prediction from the current best base learner (a vote is applied in case of ties). For simplicity $\tau = \tau_{mat}$. This simple method offers a good trade-off between keeping as much as possible relevant information about the past and be reactive when the underlying concept changes aka. the stability-plasticity dilemma.

3.2 Recognizing Recurring Concepts

Coping with recurring concepts implies, first, to be able to store memory traces of past relevant concepts and, second, to recognize which past concept is relevant again in order to exploit this knowledge.

In our approach, the memory lies entirely in the pool of base-learners. The question is then to use this pool in order to detect when a regularity deserves to be stored away and then to memorize this regularity for potential future use.

The technique we propose is based on the assumption that the base learners converge toward approximately the same, near optimal, concept, as measured

by their prediction performance, given a stationary environment and a sequence of examples of sufficient length. Assuming this, therefore, one way to detect that a stationary environment has settled is to check that the diversity of the base learners is low, under some threshold, while their prediction rate peaks.

Accordingly, we define a *stability index* that compounds a measure of diversity with an estimation of performance for the best half of the base learners in the pool[1]. We suggest the kappa statistics $\mathcal{K}$ [14] in order to compute *diversity*. This statistics measure evaluates the degree of agreement between the classification of a set of items by two classifiers[2]. In case of complete agreement, $\mathcal{K} = 1$. If there is no agreement other than what would be expected by chance, $\mathcal{K} = 0$. The *stability index* at time t is computed over the last τ_s received examples: $I_{stability} = agreement - error$ where *agreement* and *error* are computed over the best half of the current hypotheses in the pool.

$$agreement = \frac{\sum_{i=1}^{N/2} \sum_{\substack{j=1 \\ i \neq j}}^{N/2} \mathcal{K}_{h_t^i, h_t^j}}{\frac{N}{2} * (\frac{N}{2} - 1)} \tag{1}$$

and:

$$error = \frac{\sum_{j=0}^{\tau_s - 1} \sum_{i=1}^{N/2} err\left(h_t^i(\mathbf{x}_{t-j}), y_{t-j}\right)}{\tau_s * \frac{N}{2}} \tag{2}$$

where N is the size of the pool.

It is then possible to draw a curve of the successive stability indexes (see Figure 1 bottom). For each time step when the curve overcomes a given threshold, the best base learner in the pool is considered as a candidate *snapshot* (a description of the current governing concept). This snapshot is compared with the list of already stored ones and is kept only insofar as it sufficiently differs from all of them. Here again the agreement statistics can be used to measure the difference between a candidate snapshot h_t^* and each stored snapshot $h_{t_k}^*$ on the last τ_s examples. In case the agreement is less than some predefined threshold θ_d, the current candidate snapshot h_t^* is added to the list $\mathcal{M}_{LT}$ (Long Term Memory) which represents past stationary states of the environment. These snapshots are evaluated in the same way as the base learners in the pool and compete for the prediction of y_t given the current $\mathbf{x}_t$. In this way, except for a moderate overhead, the prediction performance of the system is guaranteed to be at least as good as the one of the purely adaptive strategy. (See Algorithm 1)

4 Datasets and Experiments

We carried out experiments on two artificial datasets (STAGGER and ELIST) and one real dataset (SPAM). These data sets are well-known benchmarks (see for instance [2, 5, 10, 12]).

[1] Taking into account the poorest base learners induces instabilities that lead to inferior performances.

[2] Other agreement statistics should do as well.

Algorithm 1. Selection of snapshots by ADACC.

input : The stability threshold θ_I, the difference threshold θ_d, and the evaluation window τ_s

begin
  $E_0 \leftarrow \emptyset$; `/* Ensemble of experts */`
  $\mathcal{M}_{LT} \leftarrow \emptyset$; `/* List of snapshots */`
  $k \leftarrow 0$;
  for $t = 1$ **to** ∞ **do**
    `/* Adaptation */`
    $(\mathbf{x}_t, y_t)$ is the current training instance;
    $[E_t, \tilde{y}_t] \leftarrow AdaptationEnsemble(E_{t-1}, \mathbf{x}_t, y_t)$;
    `/* Anticipation */`
    $H = \{h_t^i\}_{i=1}^{N/2}$ is the best half of experts in E_t;
    $$agr = \frac{1}{\frac{N}{2} * (\frac{N}{2} - 1)} \sum_{i=1}^{N/2} \sum_{\substack{j=1 \\ i \neq j}}^{N/2} \mathcal{K}_{h_t^i, h_t^j};$$
    $$error = \frac{1}{\tau_s * \frac{N}{2}} \sum_{j=0}^{\tau_s - 1} \sum_{i=1}^{N/2} err\left(h_t^i(\mathbf{x}_{t-j}), y_{t-j}\right);$$
    $I_{stability} = agr - error$;
    `/* Detect Stable Concept */`
    if $I_{stability} \geq \theta_I$ **then**
      $h_t^* = snapshot(E_t)$;
      `/* Detect New Concept */`
      if $isEmpty(\mathcal{M}_{LT})$ **or** $\mathcal{K}_{C_j, h_t^*} \leq \theta_d, \forall j \in [1 \ldots k]$ **then**
        $k = k + 1$;
        $C_k = h_t^*$;
        $\mathcal{M}_{LT} = add(\mathcal{M}_{LT}, C_k)$;

STAGGER [15]. This data stream corresponds to three successive target concepts: $A \Leftrightarrow size = small \wedge color = red$, $B \Leftrightarrow color = green \wedge shape = circular$ and $C \Leftrightarrow size = medium \vee large$. Each concept governs the labeling of 10,000 training instances chosen uniformly from the instance space. To simulate recurring contexts, we concatenated the original sequence with a copy of it, creating a stream of size 60,000.

ELIST [5]. This is a stream of email messages from different topics that are sequentially labeled as *interesting* or *junk* by a user. The stream contains 1,500 examples with 913 attributes (boolean bag-of-words representation). Two contexts succeed each other. In one, the user is only interested in messages related to medicine. In the other, the user's interest switches to space and baseball. The stream is the sequence C_1, C_2, C_1, C_2, C_1 where C_1 and C_2 are sequences of 300 email messages, labeled according to the first and second context respectively.

SPAM. This dataset [5] consists of 9,324 instances drawn from the email messages of the Spam Assassin Collection using the boolean bag-of-words representation with 500 attributes. As mentioned in [5], the characteristics of the spam messages gradually change as time passes. ELIST and SPAM are available in arff format at `http://mlkd.csd.auth.gr/concept_drift.html`.

4.1 Experiments

The ADACC system was evaluated against the following methods:

Simple incremental classifier (SIC): a single incremental classifier.
Moving window (MW): incrementally learns over the last w instances.
Weighted examples (WE): larger weights are assigned to recent examples in order to gradually forget the outdated information.
Dynamic Weighted Majority (DWM) [10]: an ensemble method that does not use a drift detection system. Each classifier is initially assigned a weight of one. If a classifier misclassifies an instance, its weight is multiplied by a factor $\rho < 1$. A classifier is removed if its weight falls below a threshold θ. A new classifier is added when the ensemble misclassifies an instance. The ensemble size is thus variable. The frequency of updating weights, removing and adding classifiers, is controlled by a parameter p. The decision of the ensemble is obtained by weighted majority voting.
Leveraging Bagging (LBAG) [16]: a version of online bagging that uses the ADWIN method [17] to detect concept drifts. The instances are weighted according to a Poisson(λ) distribution with $\lambda > 1$ in order to achieve more diversity in the generated weight values. When a concept drift is detected, the worst classifier is replaced with a new one. ADWIN's parameter is a confidence bound γ.
Early Drift Detection Method (EDDM) [18]: a classifier with a drift detection system. EDDM monitors the distance between consecutive classification errors d and defines two distance levels (warning and drift) with respective thresholds α, β ($\beta < \alpha$). If $d < \alpha$, the examples are stored in anticipation for change. If $d < \beta$, the classifier is reset and trained on the examples stored since the warning level.
Conceptual Clustering and Prediction (CCP) [5] : an ensemble method that uses clustering and is able to handle recurring concepts. Each batch of m instances is mapped into a conceptual vector (descriptor). The vector is either assigned to an existing cluster according to a distance threshold θ or a new cluster is created. In the latter case, a classifier trained on the batch is assigned to the cluster. In the former case, the classifier of the corresponding cluster is updated with the batch instances. The classifier of the new or existing cluster is then used to classify the next m instances. The number of clusters cannot be larger than c_{max} otherwise the new item is incorporated into the nearest cluster.
Dynamic Adaptation to Concept Changes (DACC) the mere adaptive side of the ADACC approach, as presented in Section 3.1.

4.2 Experimental Setup

Incremental Naive Bayes classifiers were used as base learners, because they naturally learn incrementally and are often used in studies of on-line learning.

LBAG and EDDM algorithms are available in the MOA (*Massive Online Analysis*) API[3]. We implemented all remaining algorithms on top of MOA, except for CCP and DWM whose results reported below are taken from the work of Katakis et al. in [5]. The parameters of DWM were set to $\rho = 0.5$, $\theta = 0.01$, $p = 1$ and those of CCP were set to $b = 50$, $c_{max} = 10$ and $\theta = 4$ for ELIST and $\theta = 2.5$ for SPAM. Both DWM and CCP were not evaluated on STAGGER in [5] and thus no results are reported on this dataset.

We evaluated MW with three different window sizes: 50, 100 and 200 (retaining the best one: 100). WE was tested with the weighting formula $w(n) = w(n-1)+n^2$ where $w(n)$ is the weight of the n-th example. The threshold values of EDDM are automatically set by MOA.

In LBAG, we tuned the parameters λ and γ experimentally converging on $\lambda = 20, \gamma = 0.002$ for ELIST and STAGGER and on $\lambda = 20, \gamma = 0.01$ for SPAM. Error correcting codes can be used for LBAG to add more diversity in the ensemble, but this does not improve the accuracy and is thus not used.

The parameters of ADACC, our *anticipative meta-learning* approach, were set to $\theta_I = 0.8$, $\theta_d = 0.7$ and $\tau_s = 100$ in all experiments. To reduce the computation cost, the anticipation mechanism is called every $p = 100$ time steps (lines 8 to 17 of Algorithm 1). Finally, the parameters of the *adaptive learning* mechanism, DACC, were optimized experimentally and $\tau_{eval} = \tau_{mat} = 20$ were used for all datasets. The ensemble size was fixed to 20 for DACC and LBAG.

4.3 Results

Table 1 reports the results averaged over 10 runs showing the accuracy, precision, recall and the run time in CPU seconds. All experiments were executed on an Intel Core i5 CPU at 2.4 GHz with 4.0 GB of RAM. The execution time of DWM and CCP are not given since they were tested in [5] on a different machine.

In all cases, ADACC yields the best accuracy. Figure 1 (top) shows a moving average of the accuracy of DACC, ADACC, LBAG and EDDM over sliding windows of 1,000 instances. DACC adapts faster to concept drifts than EDDM and LBAG, probably because of the frequent removal and addition of classifiers (every 20 time steps) which makes it ready to any upcoming change. However, despite the very good performance of DACC, ADACC still tops it by recognizing recurring concepts starting at time step 30,000. This comes at the expense of the execution time which is multiplied by a factor of 1.5 to 3. The amount of computation can be reduced by increasing the value of p, the period separating two calls of the meta-learning mechanism. In STAGGER, the run time of ADACC is reduced to 2.8 CPU seconds when $p = 1,000$ time steps (instead of 100).

[3] `http://sourceforge.net/projects/moa-datastream/`

Table 1. The accuracy, precision, and recall (in %) along with the execution time (in CPU seconds) of the different approaches on the ELIST, SPAM and STAGGER datasets using Naive Bayes classifiers as base learners.

	ELIST				**SPAM**				**STAGGER**			
Algorithm	Acc.	Precis.	Recall	Time	Acc.	Precis.	Recall	Time	Acc.	Precis.	Recall	Time
SIC	54.2	50.7	69.3	0.53	90.7	94.2	93.2	1.27	64.5	62.4	89.5	0.35
MW	74.7	70.6	78.4	0.48	90.7	90.6	97.5	1.28	98.8	98.4	99.3	0.39
WE	66.9	64.9	63.9	0.53	92.8	95.2	95.0	1.28	78.5	77.6	85.9	0.4
DWM	43.8	47	42.5	-	91.8	84.8	83.1	-	-	-	-	-
CCP	77.5	79.7	77.6	-	92.3	85.7	83.9	-	-	-	-	-
EDDM	75.6	72.9	75.9	1.15	90.8	92.0	95.9	1.68	99.7	99.73	99.8	0.6
LBAG	58.5	54.4	68.3	15.0	91.8	95.6	93.3	14.60	89.9	85.7	98.1	1.51
DACC	76.2	73.8	75.9	9.52	94.7	95.1	97.8	11.9	99.9	99.9	99.9	1.06
ADACC	**77.5**	**75.2**	**77.2**	13.6	**94.9**	**95.6**	**97.6**	18.92	**99.9**	**99.9**	**99.9**	3.22

Note that the classification performance is not hurt as long as p is small enough to take snapshots of all encountered concepts (i.e. $p < 10,000$ for STAGGER).

Figure 1 (bottom) shows the stability index on the STAGGER dataset and highlights when snapshots are stored. All data points above $\theta_I = 0.8$ are candidate snapshots (a total of 575) but only 5 are kept as relevant in the list $\mathcal{M}_{LT}$. The unstable behavior of the stability index from time step 1 to 10,000 reflects the difficulty of learning the first concept. Three different snapshots of the first concept are stored during this period, capturing different subspaces. Only one additional snapshot is stored for the first concept when it reappears (time steps 30,000 to 40,000) confirming that redundant snapshots are avoided by ADACC. The second concept is learnt more easily and only two distinct snapshots are taken. Learning the third concept corresponds to the highest stability index, suggesting a rather easy learning task. Only one representative snapshot of the third concept is stored at the end of the experiment. Regarding the other streams, a total of 8 snapshots were stored for SPAM and 4 for ELIST.

In our experiments, the threshold values for stability and conceptual equivalence were fixed. We varied their values on the STAGGER dataset to study their effect on the number of candidate snapshots, the number of stored snapshots and the classification performance of ADACC. The results are shown in Table 2. Smaller stability thresholds increase the number of candidate snapshots and thus of the stored ones but very much less significantly. Only 5 additional snapshots are stored when the threshold switches from 0.9 to 0.5. When varying the concept equivalence threshold, the number of candidates doesn't change since it is only related to the stability threshold value. The stored snapshots however evolve. Larger threshold values entice larger numbers of stored snapshots. Remarkably, changing both threshold values may impact the accuracy of the anticipative mechanism (ADACC) but never to the extent of being worse than the mere adaptive scheme (DACC).

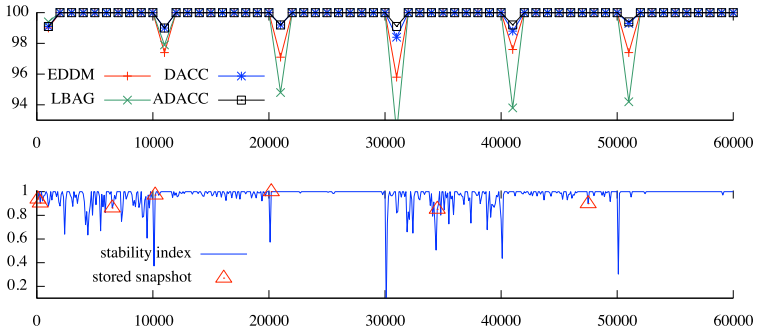


Fig. 1. (Top) The classification accuracy of ADACC, DACC, LBAG and EDDM on STAGGER, averaged over sliding windows of size 1,000. (Bottom) The evolution of the stability index on STAGGER

Table 2. *Left*: The effect of the conceptual equivalence threshold on ADACC with $\theta_I = 0.8$. *Right*: The effect of the stability index threshold on ADACC with $\theta_d = 0.7$.

θ_d	# candidates	# stored	Accuracy
0.5	575	4	99.918
0.6	575	5	99.915
0.7	575	7	99.916
0.8	575	9	99.918
0.9	575	12	99.918

θ_I	# candidates	# stored	Accuracy
0.5	596	9	99.908
0.6	594	9	99.908
0.7	587	9	99.91
0.8	575	7	99.916
0.9	540	4	99.916

5 Conclusions and Future Work

A meta-learning mechanism that deals with recurrent concepts in the context of online machine learning has been presented. The main contribution of ADACC, which explains the good performances obtained on the benchmark datasets, lies in the use of a stability measure that monitors the pool of base learners *and* the long-term memory of past useful concepts. Snapshots of the relevant states of the world are stored and re-used them when old contexts reappear. This mechanism is completely embedded in the natural functioning of the ensemble method. It relies on few parameters that do not need to be finely tuned.

We conducted experiments on real and artificial benchmark datasets and compared our method with various online learning systems. The empirical results show that combining the meta-learning mechanism with an ensemble method that adapts rapidly to drifting concepts (in comparison to other systems) can bring improvement in the classification performance, outperforming all compared systems.

In the future, we will explore ways to keep constant the size of the long term memory of the memorized snapshots. One such promising avenue is to store prototypes of snapshots instead of the original ones, using a hierarchical

clustering technique. We also plan to reduce the execution time of ADACC, and this, by computing the stability index using an agreement measure whose computational cost is less than quadratic in the number of base learners.

References

1. Gama, J.: Knowledge discovery from data streams. In: Citeseer (2010)
2. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning 23(1), 69–101 (1996)
3. Klinkenberg, R.: Learning drifting concepts: Example selection vs. example weighting. Intelligent Data Analysis 8(3), 281–300 (2004)
4. Stanley, K.O.: Learning concept drift with a committee of decision trees. In: Informe técnico: UT-AI-TR-03-302. Department of Computer Sciences. University of Texas at Austin, USA (2003)
5. Katakis, I., Tsoumakas, G., Vlahavas, I.: Tracking recurring contexts using ensemble classifiers: an application to email filtering. Knowledge and Information Systems 22(3), 371–391 (2010)
6. Yang, Y., Wu, X., Zhu, X.: Mining in anticipation for concept change: Proactive-reactive prediction in data streams. Data Mining and Knowledge Discovery 13(3), 261–289 (2006)
7. Alippi, C., Boracchi, G., Roveri, M.: Just-In-Time Classifiers for Recurrent Concepts. IEEE Transactions on Neural Networks and Learning Systems (to be published, 2013)
8. Gama, J., Kosina, P.: Tracking recurring concepts with meta-learners. In: Lopes, L.S., Lau, N., Mariano, P., Rocha, L.M. (eds.) EPIA 2009. LNCS, vol. 5816, pp. 423–434. Springer, Heidelberg (2009)
9. Gomes, J.B., Sousa, P.A., Menasalvas, E.: Tracking recurrent concepts using context. Intelligent Data Analysis 16(5), 803–825 (2012)
10. Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: A new ensemble method for tracking concept drift. In: Third IEEE International Conference on Data Mining, ICDM 2003, pp. 123–130 (2013)
11. Tsymbal, A., Pechenizkiy, M., Cunningham, P., Puuronen, S.: Dynamic integration of classifiers for handling concept drift. Information Fusion 9(1) (2008)
12. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., Gavaldà, R.: New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2009)
13. Scholz, M., Klinkenberg, R.: An ensemble classifier for drifting concepts. In: Proceedings of the Second International Workshop on Knowledge Discovery in Data Streams, Porto, Portugal (2005)
14. Carletta, J.: Assessing agreement on classification tasks: the kappa statistic. Computational Linguistics 22(2), 249–254 (1996)
15. Schlimmer, J.C., Granger Jr., R.H.: Incremental learning from noisy data. Machine Learning 1(3), 317–354 (1986)
16. Bifet, A., Holmes, G., Pfahringer, B.: Leveraging bagging for evolving data streams. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010, Part I. LNCS, vol. 6321, pp. 135–150. Springer, Heidelberg (2010)
17. Bifet, A.: Adaptive learning and mining for data streams and frequent patterns. ACM SIGKDD Explorations Newsletter 11(1), 55–56 (2009)
18. Baena-Garca, M., del Campo-vila, J., Fidalgo, R., Bifet, A., Gaval, R., Morales-Bueno, R.: Early drift detection method. In: Fourth International Workshop on Knowledge Discovery from Data Streams (2006)

Control of an Inverted Pendulum Using the NeuraBase Network Model

Robert Hercus, Kit-Yee Wong, See-Kiong Shee, and Kim-Fong Ho

Neuramatix Sdn Bhd, Kuala Lumpur, 59200 Malaysia
{Hercus,SeeKiong,KitYee,kfho}@neuramatix.com

Abstract. This paper presents an alternative approach for the control and balancing operations of an inverted pendulum. The proposed method uses a neuronal network called NeuraBase to learn the sensor events obtained via a rotary encoder and the motor events controlling a stepper motor, which rotates the swinging arm. A neuron layer called the controller network will link the sensor neuron events to the motor neurons. The proposed NeuraBase network model (NNM) has demonstrated its ability to successfully control the balancing operation of the pendulum in the absence of a dynamic model and theoretical control methods. The controller also demonstrated its robustness in the adaptive learning of pendulum balancing with imposed system changes.

Keywords: Adaptive Control, Real-time Control, Online Control, Neural Network, Inverted Pendulum, Balancing Control.

1 Introduction

The inverted pendulum is a classic problem in non-linear control systems. Much research has been made to solve this problem as the balancing control of an inverted pendulum forms the basis of many diverse phenomena such as walking, aircraft roll control and planar robot arm control. The inverted pendulum has been widely used as a platform for testing the efficacy of various types of controllers [1-12]. However, the controller design of an inverted pendulum is difficult due to its multi-variability and inherent instability.

There are generally two categories of controllers, namely, dynamic modeling and control, as well as, machine learning. In dynamic modeling and control, controllers using the methods of Proportional-Integral-Derivative (PID), pole placement, Linear-Quadratic-Regulator (LQR) and Linear-Quadratic-Gaussian (LQG), have been previously applied to the control of inverted pendulum systems [1-4]. Although most of these approaches achieved good balancing performance, an accurate mathematical representation for the inverted pendulum is not easily formulated, as extensive knowledge of the system dynamics is required. Machine learning approaches such as fuzzy logic, artificial neural networks, neuro-fuzzy, self-organizing maps, recurrent neural networks, cerebellar model articulation controller (CMAC) and genetic algorithms have also been studied extensively [5-12]. The abilities of these machine-learning methods in mapping non-linearity, and in dealing with uncertainties in system parameters, eliminate the need for exact mathematical models.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 605–615, 2013.

In this paper, the balancing control of an inverted pendulum using a temporal-based neural network model named NeuraBase [13] is introduced. The proposed NeuraBase controller does not require the system's mathematical model as its online training attribute allows it to continuously adapt to a changing environment. The same controller has been previously applied in the balancing control of a simulated inverted pendulum [14] as well as the navigation control of UAV [15]. The reinforced learning method used in this work is similar to the unsupervised spike-based Hebbian learning method presented in [16] in term of updating the strength of controller's connection between the sensor and motor neurons. In this paper, the implementation is not intended to make direct comparisons with other pendulum balancing methods, but to introduce an alternative and novel approach. The NeuraBase generic toolbox can be downloaded at [17].

As shown in Figure 1, the rotary inverted pendulum consists of a pivot arm rotating in a horizontal plane by means of a motor. At the other end of the arm, a pendulum is mounted, rotating in a plane that is always perpendicular to the rotating arm. (The swing-up process necessary to bring the pendulum to the balancing region of 0 ± 10° at the outset was controlled using another NeuraBase controller, and will not be discussed in this paper.)

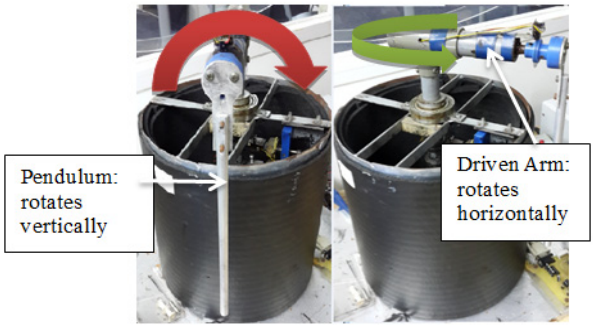


Fig. 1. The rotary inverted pendulum

This paper is organized as follows: Section 2 of this paper describes the usage of the NeuraBase Network Model (NNM) as a controller. In Section 3, the model of the inverted pendulum and the experimental set up is described. Section 4 describes the online learning logic used with the NNM, and in section 5 experimental results and discussions are presented.

2 NeuraBase Network Model (NNM)

The NNM is a network data structure that stores a sequence of events. The basic unit of the NNM is a neuron. Each neuron represents an event. Two neurons can be joined to represent a sequence of sensor or motor events. As shown in Figure 2a below, the neurons in the NNM can be associated temporally or spatially. These events are constructed in a way that provide for fast searching and matching. For instance, as shown in Figure 2b, if each alphabet is represented by a sensor neuron (Level 1), the association of overlapping sensory events can represent words.

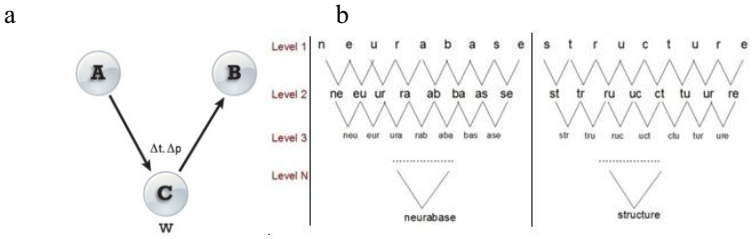


Fig. 2. a) The NeuraBase Network Model where *t* denotes time proximity and *p* denotes spatial proximity; b) Words as sequences of events constructed using characters

The proposed NNM controller for the inverted pendulum consists of three different networks as shown in Figure 3. For the inverted pendulum, both the motor and controller networks have a single level architecture while the sensor network has a multi-level architecture. These three networks store different types of events, namely, a) sensor events - input to the system (the pendulum position readout from the encoder); b) motor neurons - outputs from the system (the motor velocity change for driving the arm); c) controller neurons - associations between the sensor events and motor neurons. Each type of event builds an association of events in their respective network. The sensor network, motor network and the controller network store sensor neurons events, motor neurons events, and controller neurons associations, respectively. A simplified data structure of the neurons used in NNM is described in Table 1. More detailed descriptions of the sensor, motor and controller neurons are provided in Section 3.

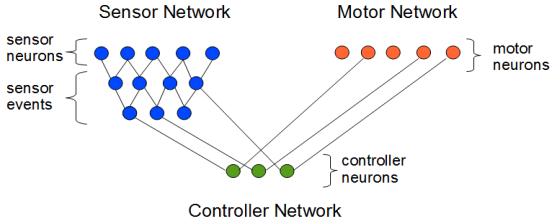


Fig. 3. The architecture of the network of NeuraBase used in the balancing of the inverted pendulum

Table 1. Data structure of a neuron (basic), * denotes fields that are only applicable to the controller neuron

Field	Data Type
Head	unsigned int
Tail	unsigned int
Successor	unsigned int
Frequency/ Weight*	signed int
Next	unsigned int
Overshoot/Undershoot Flags*	unsigned short

3 Experimental Setup

The experimental set-up of the inverted pendulum is depicted in Figure 4a. The hardware system of the inverted pendulum consists of two main components, namely, the rotating arm and the pendulum. The actuator that drives the rotating arm is a stepper motor with a 1.8˚ step angle. For the sensor, an encoder plate with a resolution of up to 0.125 degrees is attached to the pendulum.

As shown in Figure 4b, an encoder is installed at the pivot point of the pendulum to calculate the angular position. The resting position of the pendulum, pointing to the ground, is the starting point or 180°, and the upright position of the pendulum, pointing to the ceiling, is the balancing point or 0°.

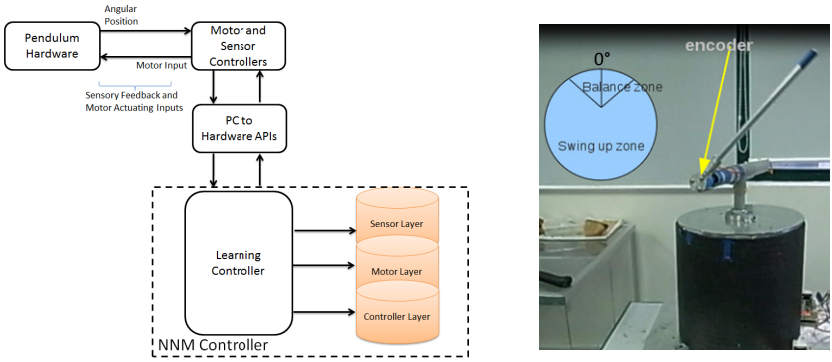


Fig. 4. a) Experimental setup of the inverted pendulum system; b) Illustration of the balance and swing-up zones, and the position of the encoder

The figure also depicts how the angular measurement of the pendulum is defined. The angular positions of the pendulum are determined according to the segmentation scheme, wherein the full circle in the plane perpendicular to the rotating arm is segmented into 72 discrete segments. Further descriptions on the segmentation scheme are given in the next section. In order to ensure high speed monitoring of the encoder readout, the sensor controller communicates with the PC at a baud rate of 57,600bps. The sensor controller firmware is optimized to efficiently handle the encoder high pulse rate of up to 14 kHz, which is approximately 5 turns per second.

A segmentation scheme is used to represent the defined range of angles for the pendulum positions. As shown in Figure 5 below, the segments are defined by the pendulum's position with reference to the upright (0°) position. The operating region for the balancing mode is set at $\pm 10°$ from the upright position of 0°. The angular measurement is divided into two symmetrical sides, whereby, positive segments represent the right side and negative segments represent the left side. There are, altogether, 21 segments within the balancing zone; hence, there are only 21 basic sensor neurons underlying a sequence of sensor events for the inverted pendulum balancing model.

For motor neurons, changing the velocity between two time samples controls the stepper motor rotation. Each motor neuron $\boldsymbol{M}$ is represented as a unit velocity change,

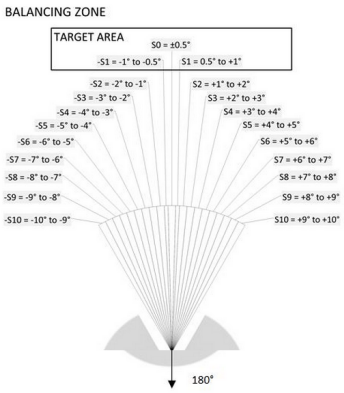


Fig. 5. The segmentation of angular positions of the inverted pendulum in the balancing zone

which is then applied to the arm rotation, where each unit corresponded to a shift of $0.02\pi rad/s$, with either a positive or negative sign representing counter-clockwise or clockwise rotation respectively. With a change limit of $\pi rad/s$ imposed in either direction($\pm M50$), there are consequently 101 basic motor neurons defined to represent these changes in arm velocity.

The controller neurons associate sensor event sequences with motor neurons. The association indicates the possible variables to control the pendulum through a learning process. The inverted pendulum problem is proposed to be solved using the learning model - given a sequence of angular positions (***S***) of the pendulum, a change in motor speed or arm velocity (***M***) can be applied to move the pendulum to the desired balance position. The relationship between the sensor events and the motor neurons are associated (linked) via a controller neuron. At the frequency of seventy times per second, ***S*** was obtained and stored in the sensor network, forming a sequence of position segments. Similarly, each ***M*** defined for the motor was stored in the motor network.

Figure 6a depicts the neuron structure of a sample sensor event *C,* within the sensor network, which represents the sequence of positions $C = \{S6, S4, S1\}$.

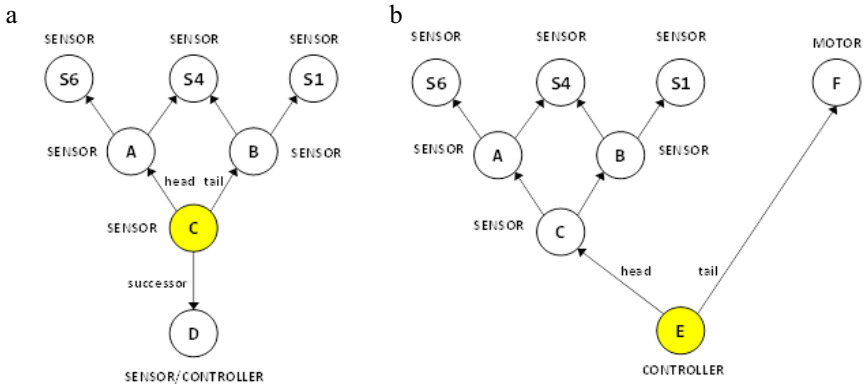


Fig. 6. a) The structure of a sensor event; b) The structure of a controller neuron

The neuron *A* (head of *C*) represents the sensor sequence of $\{S6, S4\}$ and the neuron *B* (tail of *C*) represents the sensor sequence of $\{S4, S1\}$. Neuron *D* represents a set of successors of *C,* which can be a sequence of position segments e.g. $\{S6, S4, S1, S0\}$ etc. or *D,* could be controller neuron for *C,* linking to a specific motor action.

Each motor neuron represents a unit velocity change to the arm rotation. Figure 6b depicts the neuron structure of a sample controller neuron (*E*) within the controller network. The neuron *C* (head of *E*) is the sensor event mentioned above and the neuron *F* (tail of *E*) is a motor neuron.

4 Learning Logic

Using the reinforcement learning method, the goal of the NNM controller is to bring the pendulum into the target region $0 \pm 1°$ or $[-S1, S1]$ and maintain the pendulum in the target region for as long as possible. The proposed learning method is different from supervised learning, which depends strongly on the availability of an external teacher, whereby, the system output is assessed according to the desired output given by the external teacher.

The direction of the motor speed change applied in order to balance the pendulum was based on the following learning rule: whenever the inverted pendulum falls out of the target $[-S1, S1]$ region, the NNM controller will attempt to predict the best motor action to execute, by searching within the controller network for the strongest motor actions exceeding a predefined weight, which will bring the pendulum into the target area $[-S1, S1]$. The sensor sequence events have a fixed maximum length of *n*, which is set to 5. For instance, given a sequence of sensor events $\{-S3, -S2, -S2, -S1, -S1\}$, there may be three trained controller neurons $N3, N4, N5$ which correspond to motor neurons -2, -1, and 1, respectively. The strength of each controller neuron is represented by its trained weight; hence, the controller neuron with the highest weight will be the most reliable prediction because it is the accumulated result of learning, using both positive and negative feedbacks from past trials.

The association of a sensor event (***S***) and a motor neuron (***M***) in the controller network is a reinforced learning process, whereby positive and negative feedbacks dictate how the learning takes place by tuning the weight of the controller neurons. The association of the sensor event and the motor neurons in the controller network essentially gives rise to a learned inverse model that can generate motor actions given the desired sensor event sequence. Based on a predefined learning goal, the positive and negative feedback rules are defined based on whether the goal is achieved. The feedback rule is constructed according to the strategy that, given a sequence of sensor events collected using a fixed time interval (e.g. pendulum traversal path from $S3 \rightarrow S1 \rightarrow S0 \rightarrow -S1 \rightarrow -S1$*)* and a motor neuron effecting a change in speed (e.g. *M3*) has been applied, the current sensor data is $-S1$. This represents the segment position of the pendulum after the speed change (*M3*) has been applied.

A positive feedback is given if the pendulum managed to reach within the target region of $[-S1, S1]$, upon which, the weight of the controller neuron is incremented by 1. The more the controller neuron experiences positive feedbacks for its motor predictions, the stronger the link coupling will be, thereby resulting in a stronger positive memory of the respective motor action. A negative feedback is evoked if the

pendulum fails to reach the target region $[-S1, S1]$, upon which, the weight of the controller neuron is decremented by 1, thus reducing the coupling strength of that link. Alternatively, if a controller neuron does not already exist, the NNM controller will create a controller neuron linking the sensor event to the motor action executed. Eventually, a network of almost all possible sensor events associated with possible motor actions will be stored within NeuraBase, and the controller neurons linking sensor events to the right motor actions will have higher weights compared to those linking sensor events to incorrect motor actions.

In the case where there are no controller neurons with strength exceeding the weight, which is common at the beginning of training, the controller will set the sensor sequence to its tail (meaning it will check recursively for the latest *n-1, n-2,… 1* events for any matches). If none can be found even for the shortest sensor event, a random motor neuron within a reducing range will be chosen.

This motor range is bounded by motor neurons linked to previously used controller neurons which, have either been flagged as overshooting and / or undershooting controller neurons according to the rules outlined in Table 2. Each time a controller neuron's overshoot / undershoot flag is set, the set range becomes smaller with the controller neuron's linked motor neuron as the new upper or lower boundary of the set. This method helps the controller narrow down its choices of approximately correct motor actions for the sensor event more quickly, compared to the controller having to attempt all motor neurons before finding a suitable one.

Table 2. Controller neuron overshoot / undershoot flagging rules

Pendulum position		Flag
Before: [-S10,-S2]	After: [S2,S10]	Overshoot
Before: [S2, S10]	After: [-S10,-S2]	Overshoot
Before: [-S10,-S2]	After: [-S10,-S2]	Undershoot
Before: [-S10,-S2]	After: [-S10,-S2]	Undershoot
Before: [-S10,-S2]	After: [-S1,S1]	None
Before: [S2,S10]	After: [-S1,S1]	None

5 Results and Discussions

The experiment was run for a total of 90 trials, which recorded a total of $\sim 2.7\ x\ 10^6$ *motor events*, with each *motor event* representing an event where a motor action is executed as the pendulum falls out of the target balancing area$[-S1\ S1]$. Experimental results of the pendulum balancing performance (see Figure 7) show that the pendulum was able to balance for longer duration after training.

In the same figure, the *Total* line depicts the growth trend of total neurons in NeuraBase, which correlates with the upward trend of the average pendulum balancing time. Also shown are the neuron growth trends for each level of the sensor network (higher levels indicate longer sensor event sequences), and the controller network (the neuron count of the motor network is negligible). Initially, the balancing durations were generally less than 10 minutes. This is to be expected, as at the commencement of training, the NNM did not contain any stored patterns. The default motor actions $\pm M10$ may not have been sufficient for all sensor events, therefore it would take time for NeuraBase to learn the correct motor actions.

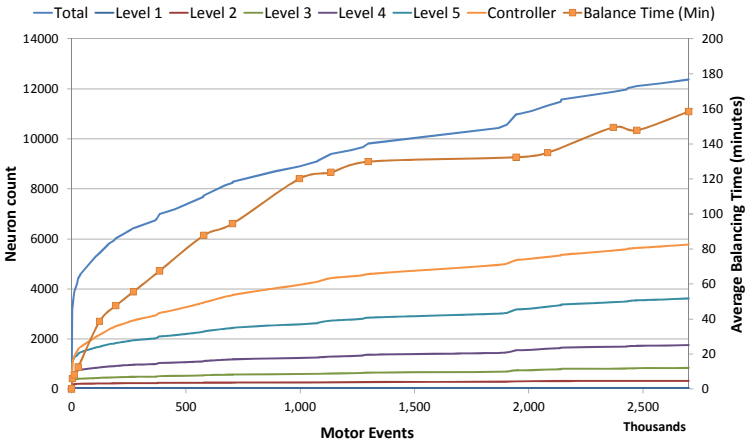


Fig. 7. Average pendulum balancing time and NeuraBase node growth vs the number of NeuraBase motor events

Additionally, the pendulum's stability does not only depend on the immediate motor action but also on all previous motor actions. All of which, have not been learned extensively, which also contributed to the short balancing times. Many of the new patterns (new neurons) were created during this period. After approximately 30,000 motor events (~40 trials), the results showed that the NNM controller was able to balance the pendulum up to the 10 minutes mark. At this point, the neuron count almost reached the 5,000 mark. After that, the controller was able to learn relatively faster (more rapidly growing average balancing time), and the NNM was able to balance the pendulum at an average of 150 minutes after 90 trials and $\sim 2.7\ x\ 10^6$ motor events later. This apparent improvement in performance can be attributed by the fact that, once the NNM controller succeeded in bringing the pendulum into the target region and maintaining it for a few time steps, more and more motor events occurred from frequently repeating sensor events. Therefore, approximately correct motor actions would have already been learned and the controller was able to sustain its balance more easily, at which point, the pendulum can be considered to be stable.

The growth trend of NNM neurons had yet to reach saturation point after 90 trials ($\sim 2.7\ x\ 10^6$ motor events), as the NNM was still learning new patterns, whilst balancing the pendulum. Even in supposed stability, the pendulum did not remain static at the 0° point. Learning continued to occur whenever the pendulum fell outside the target $[-S1\ S1]$ region, during which the NNM controller may encounter new sensor events and / or learns new motor actions or strengthens / penalizes current motor actions in order to bring the pendulum back into the $[-S1\ S1]$ region. As a result, the total number of cumulative motor events was observed to be increasing exponentially with respect to the number of trials, as shown in Figure 8. A video, which demonstrates our experiment can be found in http://www.neuramatix.com/video_inverted.php

An additional experiment was performed to further demonstrate the effectiveness of the proposed NNM as an adaptive controller using varying weights for the tip

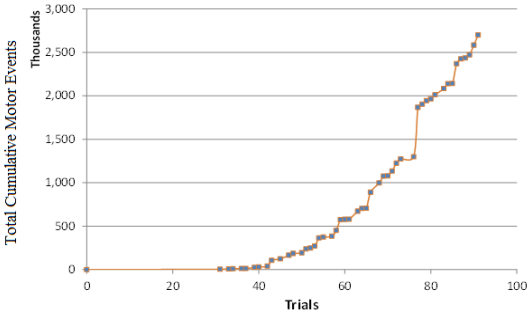


Fig. 8. Total cumulative motor events vs number of the inverted pendulum balancing trials in NeuraBase

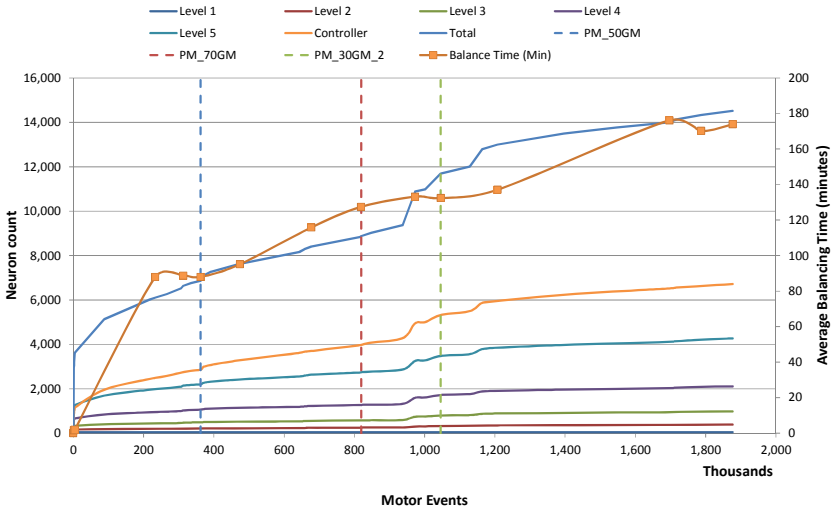


Fig. 9. Average balancing time (minutes) vs. number of motor events using three different weights of tip mass trained on the same NeuraBase (orange line with markers)

mass, which was attached to the pendulum rod. In this test, the inverted pendulum was initially trained with an empty NNM. Three different weights (70, 50 and 30 grams) were attached to the pendulum at the same distance from the pivot to introduce different three centers of mass.

In Figure 9, the vertical blue, red and green lines denote the number of motor events when the weight of the tip mass is changed from 30 gram → 50 gram → 70 gram → 30 gram, respectively. After 50 training trials (~380,000 motor events) with the 30 gram tip mass, the inverted pendulum was replaced with a heavier weight, which lowered the center of mass of the system. The NNM that was previously trained using the lighter weight was put into use and run for another 8 trials (~460,000 motor events). Another change was introduced to the pendulum, whereby the tip mass was changed to an even heavier weight. The tip mass was replaced with the original weight of 30 gram after a further training of another 6 trials (~226,000 motor events).

It is noted that the reduced amount of motor events recorded as the tip mass was changed indicates that the pendulum was generally able to spend more time in the target $[-S1\ S1]$ area after the motor action has been executed by the NNM controller, and does not have any significant meaning, as 'time spent in the target area' was not part of the learning feedback rules. The orange line with markers represents the performance of the NNM controller (average balancing time) trained using a single NNM with three different top masses continuously in real-time. Results showed that the NNM controller was robust in handling changes to the system as the balancing persists even when changes have been made twice to the system.

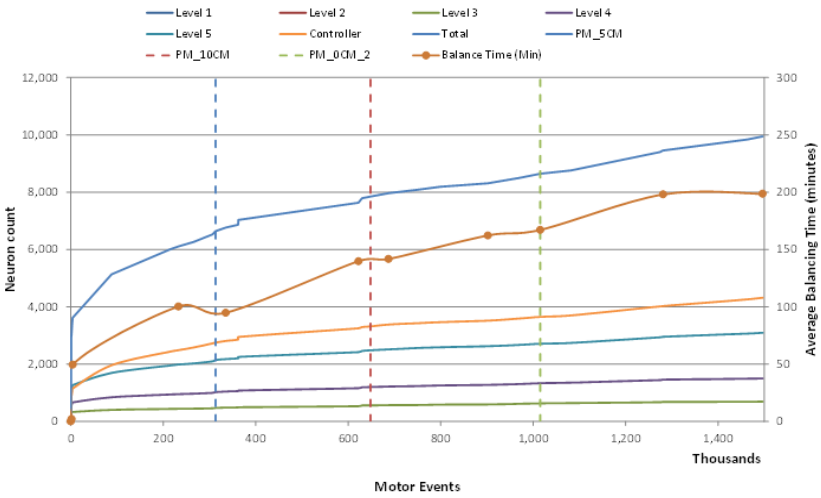


Fig. 10. Average balancing time (minutes) vs. number of motor events using three different centers of mass trained on the same NeuraBase (orange line with markers)

In another separate experiment, a 50-gram tip mass was attached to the pendulum at three different positions measured from the pendulum pivot to introduce three centers of mass. The inverted pendulum was trained with an empty NNM. Figure 10 depicts the balancing performance of the NNM controller and the experimental results showed that the NNM controller was well equipped to balance the pendulum in the event of system changes.

6 Conclusion

The self-learning NNM controller presented in this paper is a proof of concept that the NNM can be easily adapted to handle the classic control problem for the balancing operation of an inverted pendulum. The experiment's results showed that the NNM controller was able to learn to balance the pendulum. The learning element was evidenced by the increasing average balancing time and growth trend of NeuraBase nodes. The proposed NNM controller also demonstrated its robustness in adaptively learning to balance the pendulum in the event of system changes.

References

1. Mertl, J., Sobota, J., Schlegel, M., Badal, P.: Swing-up and Stabilization of Rotary Inverted Pendulum. In: Proceedings of Process Control, pp. 1–6. Slovak University of Technology (2005)
2. Sukontanakarn, V., Parnichkun, M.: Real-time Optimal Control for Rotary Inverted Pendulum. American Journal of Applied Sciences 6(6), 1106–1115 (2009)
3. Astrom, K.J., Furuta, K.: Swinging up a Pendulum by Energy Control. Automatica 36(2), 287–295 (2000)
4. Nasir, A.N.K., Ahmad, M.A., Rahmat, M.F.: Performance Comparison between LQR and PID Controller for an Inverted Pendulum System. In: Proceedings of International Conference on Power Control and Optimization (2008)
5. Radhamohan, S.V., Subramaniam, M., Nigam, M.J.: Fuzzy Swing-up and Stabilization of Real Inverted Pendulum using Single Rulebase. Journal of Theoretical and Applied Information Technology, 43–50 (2005)
6. Minnaert, E., Hemmelman, B., Dolan, D.: Inverted Pendulum Design with Hardware Fuzzy Logic Controller. Journal of Systemics, Cybernetics and Informatics 6(3), 34–39 (2008)
7. Hayashi, I., Nomura, H., Wakami, N.: Acquisition of inference rules by neural network driven fuzzy reasoning. Japanese Journal of Fuzzy Theory and Systems 2(4), 453–469 (1990)
8. Zheng, Y., Luo, S., Lv, Z.: Control Double Inverted Pendulum by Reinforcement Learning with Double CMAC Network. In: Proceedings of The 18th International Conference on Pattern Recognition, vol. 4, pp. 639–642 (2006)
9. John, D.H., Fischer, J., Johnam, D.: A Neural Network Pole Balancer that Learns and Operates on a Real Robot in Real Time. In: Proceedings of the MLC-COLT Workshop on Robot Learning, pp. 73–80 (1994)
10. Tatikonda, R.C., Battula, V.P., Kumar, V.: Control of Inverted Pendulum using Adaptive Neuro Fuzzy Inference Structure. In: IEEE Internal Symposium on Circuits and Systems, pp. 1348–1351 (2010)
11. Tetsuya, M., Furukawa, T.: The Self-Organizing Adaptive Controller. International Journal of Innovative Computing. Information and Control 7(4), 1933–1947 (2011)
12. Lin, C.J., Lee, C.Y.: Non-linear System Control using a Recurrent Fuzzy Neural Network based on Improved Particle Swarm Optimisation. International Journal of Systems Science 41(4), 381–395 (2010)
13. Hercus, R.G.: Neural networks with learning and expression capability. U. S. Patent 7412426 B2 (2008)
14. Hercus, R., Wong, K.-Y., Ho, K.-F.: Balancing of a Simulated Inverted Pendulum using the NeuraBase Network Model. In: Mladenov, V., Koprinkova-Hristova, P., Palm, G., Villa, A.E.P., Appollini, B., Kasabov, N. (eds.) ICANN 2013. LNCS, vol. 8131, pp. 527–536. Springer, Heidelberg (2013)
15. Hercus, R., Kong, H.-S., Ho, K.-F.: Control of an Unmanned Aerial Vehicle Using a Neuronal Network. In: Conf. Proc. IEEE Symposium Series on Computational Intelligence (2013)
16. Wang, X., Hou, Z.-G., Zou, A., Tan, M., Cheng, L.: A Behavior Controller Based on Spiking Neural Networks for Mobile Robots. Neurocomputing 71, 655–666 (2008)
17. NeuraBase Generic Toolbox, `http://neuramatix.com/`

On the Analysis of Time-Delayed Interactions in Genetic Network Using S-System Model

Ahsan Raja Chowdhury[1,2], Madhu Chetty[1,2], and Nguyen Xuan Vinh[1]

[1] Gippsland School of Information Technology, Monash University, Australia
[2] National Information and Communication Technology Australia (NICTA)
{ahsan.chowdhury,madhu.chetty}@monash.edu, vinh.nguyenx@gmail.com,
{ahsan.chowdhury,madhu.chetty}@nicta.com.au

Abstract. The Gene Regulatory Network (GRN) is the collection of genes and interactions among them, which captures the mutual interactions among genes. Amongst the various currently available models for inferring GRN, the S-System formalism is often considered as an excellent compromise between accuracy and mathematical tractability, although limited to represent the instantaneous interactions only. Recently proposed Time-delayed S-System Model (TDSS), an improved version of the traditional S-System model, is capable of representing the delayed interactions present in the genetic network. In this paper, we have shown the results of extensive analysis performed on TDSS over a widely used synthetic network. The two well-known performance measures applied to the synthetic network with various time-delayed regulations clearly demonstrate that the TDSS can capture both the instantaneous and delayed interactions correctly with high precision. Further, we have shown the effect of various samples sizes during the optimization where average error for the inferred parameters are reported and compared with an existing state-or-the-art algorithm.

Keywords: Gene Regulatory Network, S-System, Time-delay.

1 Introduction

Reverse engineering GRN with the S-System model carry out the attraction of the researchers in the last decade due to its originality in the applications. Despite of the large number of parameters, the S-System formalism has great ability to cope with non-linearity and to infer GRNs more accurately than the other available models [1–5]. However, the applications of the S-System in GRN are limited to the reverse engineering of small scale genetic networks due to very high computational complexity. Gradually, these limitations were overcome by the decomposition technique and parallel computation, which were found useful in reconstructing medium scale networks [2, 4], although inappropriate for large scale network.

In the biological system, almost all genetic interactions in GRNs are invariably delayed with different time lags [6]. An interaction among two genes is said

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 616–623, 2013.

to be instantaneous if it occurs in the same time-sample/timestamp (TS). On the other hand, if the interaction from the source gene starts in one TS and ends in a later TS of another gene is known as time-delayed interaction. In the currently existing S-System model based reconstruction techniques, that use the time-series data, the interactions are by default assumed to be instantaneous. Hence, the delayed interactions, if present in the network, are missed or inferred with incorrect regulatory weight by the reconstruction method. Being cognizant about the limitations of the existing S-System based modeling approaches, our recently proposed Time-delayed S-System Model (TDSS) [7], in contrast to other non-S-System based time-delayed modeling approached [8,9], is capable of simultaneously inferring both the instantaneous regulations and time-delayed regulations of any fractional delay present in a genetic network. This paper presents the evaluation of the proposed TDSS for extensive analysis performed over different delay networks with variations in number of samples in the time-series data. The average errors incurred by the TDSS for various networks are also reported and compared with recently proposed S-System based method REGARD [10]. Unlike the existing TDSS proposed in [7], we have kept the delay parameters as fixed throughout the optimization and performances are observed in order to retrieve the regulatory interactions (both delayed and instantaneous). We observe that, with the known delay parameters, TDSS can infer the regulations more quickly than learning those parameters in the optimization.

2 Preliminaries on S-System Model

For a network of N genes, the S-System model is given by:

$$\frac{dX_i}{dt} = \alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}}, \quad i = 1 \ldots N \tag{1}$$

Here, for any i^{th} gene, X_i is the expression level, $\{\alpha_i, \beta_i\} > 0$ are the rate constants, $\{g_{ij}, h_{ij}\}$ are the kinetic orders (generally $-3.0 \leq \{g_{ij}, h_{ij}\} \leq 3.0$ in regular biochemical systems [5]). $g_{ij} = 0$ implies that there is no regulation in production from gene j on gene i. If $g_{ij} > 0$, gene j activates gene i and if $g_{ij} < 0$, gene j inhibits gene i. In contrast to g_{ij}, the term h_{ij} represents an opposite effect of the gene j on i for degradation. To infer a GRN of N genes using the S-System model, $2N(N+1)$ parameters must be estimated. To overcome high computational complexity, a decoupled system [1] divides the given problem into N sub-problems, where each of having $2(N+1)$ parameters are estimated by solving decoupled S-System equation. Although the accuracy may slightly decrease due to direct estimation rather than numerical integration of time series data, this approximation greatly reduces the computational burden. However, the S-System model (Eqn. (1)) calculates the change of expression for each gene depending on the current expression levels of all genes', which implies that traditional S-System model works on instantaneous regulations only.

3 The Representation Framework

In this section, we introduce the Time-delayed S-System model [7], with the parameter learning mechanism applied in this paper for reverse engineering GRN.

3.1 Time-Delayed S-System (TDSS): The Model

It is already mentioned that, for a particular gene (gene-i) in an N gene network, the S-System can represent $2N$ interactions with N interactions of production phase and remaining of degradation phase [5]. As a result, delay can occur in any of the $2N$ interactions. Thus, we propose the Time-delayed S-System (TDSS) [7] model in the following way (canonical version):

$$\frac{dX_{i,t}}{dt} = \alpha_i \prod_{j=1}^{N} X_{j,(t-\tau^g_{i,j})}^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_{j,(t-\tau^h_{i,j})}^{h_{ij}} \quad i = 1 \dots N, t = 0 \dots T \tag{2}$$

Here, T being the number of samples in the time-series data. τ^g and τ^h are delay matrices represent the delays (or lags) for g and h metrics, respectively. The delay matrices are represented as follows:

$$\tau^g = \begin{pmatrix} \tau^g_{1,1} & \tau^g_{1,2} & \cdots & \tau^g_{1,N} \\ \tau^g_{2,1} & \tau^g_{2,2} & \cdots & \tau^g_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \tau^g_{N,1} & \tau^g_{N,2} & \cdots & \tau^g_{N,N} \end{pmatrix} \tau^h = \begin{pmatrix} \tau^h_{1,1} & \tau^h_{1,2} & \cdots & \tau^h_{1,N} \\ \tau^h_{2,1} & \tau^h_{2,2} & \cdots & \tau^h_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \tau^h_{N,1} & \tau^h_{N,2} & \cdots & \tau^h_{N,N} \end{pmatrix} \tag{3}$$

The values in $\tau^g_{i,j}$ ($\tau^h_{i,j}$) represent the corresponding delay for the interaction of gene-j on gene-i in the production (degradation) phase, represented in g (h) metric with $g_{i,j}$ ($h_{i,j}$). These two metrics are calculated according to the following equations:

$$\tau^g_{i,j}(\tau^h_{i,j}) = \begin{cases} d \text{ if } g_{i,j} \ (h_{i,j}) \text{ is a delayed interaction of } d \text{ time, where } 0 < d \leq \tau_{max} \\ 0 \text{ if no interaction from } j \text{ to } i \text{ in the production (degradation)} \\ \quad \text{or } g_{i,j} \ (h_{i,j}) \text{ is an instantaneous interaction} \end{cases} \tag{4}$$

Here, τ_{max} is the maximum allowed delay in the GRN. In [7], along with the S-System parameters (i.e., α, β, g, h), the delay parameters (i.e., τ^g, τ^h) are also learnt during the optimization phase. However, in this paper we have considered the delay metrics as known and kept as fixed during the optimization.

3.2 Reverse Engineering GRN with TDSS

The Trigonometric Differential Evolution (TDE) method, a class of Differential Evolutionary Algorithm, has been employed here to reverse engineer the genetic network with the new TDSS model. This method begins with a knowledge based

papulation initialization algorithm [10]. In order to evaluate a solution, we have used the fitness function of REGARD [10], as shown in Eqn. (5):

$$f_i^{MSE} = \sum_{k=1}^{M} \sum_{t=1}^{T} \left\{ \frac{X_{k,i}^{cal}(t) - X_{k,i}^{exp}(t)}{X_{k,i}^{exp}(t)} \right\}^2 + C_i \frac{2N}{Z_{Count}} \tag{5}$$

Here, Z_{Count} is the total number of non-regulations for the i^{th} gene (= $2N$-total regulations) and, C_i is the scaling factor for the i^{th} gene calculated based on adaptive regulatory genes cardinality. Iteratively, crossovers, mutations and flip-operations [10] of TDE [4] within a population are performed and the best fitting individuals are taken in to account for the future generation. Within the optimization, we have used the Hill Climbing Local Search (HCLS) [4] to randomly selected 10% individuals and a flip operation [10] over 5% individuals in the first 200 generations for better mating of parameters. In this paper, we have evaluated the performances of TDSS [7] for the fixed and known delays, hence we have limited the experiment to reconstructing the synthetic networks only.

4 Experimental Results and Discussions

The performance of the proposed technique is studied by investigating the 5-gene synthetic network of [1]. This 5-gene synthetic network, shown in Table 1, was first reported in [1] and later studied with other S-System and non S-System based approaches for GRN modeling [2–4, 11, 12]. For comparison, we have considered the well-known performance measures, namely, sensitivity (S_n), specificity (S_p) and average error (AE) [10]. Two different cases of the 5-gene synthetic network are considered for evaluation: Case 1, that includes 5 delayed regulations of equal and integer delay, and Case 2, that has 6 delayed regulations of various fractional units. The delay matrices for Case 1 and Case 2 are shown in Eqn. (6) and Eqn. (7), respectively. Equation (2) has been used in the benchmark synthetic network with 10 and 20 random initial conditions to produce 10 and 20 data sets, respectively. As mentioned earlier, unlike our recently proposed technique in [7], the delay parameters are assumed to be fixed during the optimization. For each gene-i, the maximum in-degree I was chosen as 3 while the minimum in-degree J was set to the number of delayed regulations for gene-i. These in-degree values are updated using Adaptive Regulatory Genes Cardinality algorithm [7, 10], applied every 40 iterations. The parameter values for the TDE algorithm were set to $F_o = 0.5$ (Mutation Factor), $F_t = 0.05$ (Trigonometric Mutation Factor), $CF = 0.8$ (Crossover Factor), population size = 100. For these two types of cases, average errors reported for each cases (Case 1 and Case 2) according to the equation $E = \frac{1}{QP} \sum_1^Q \sum_1^P |d_1 - d_2|$, where Q=number of runs (=5), P=number of parameters, d_1 and d_2 are the target and inferred parameter values, respectively. We execute both TDSS and REGARD [10] for 400 iterations in each run.

Table 1. Target parameters for 5-gene Synthetic Network [1]

i	α_i	$g_{i,1}$	$g_{i,2}$	$g_{i,3}$	$g_{i,4}$	$g_{i,5}$	β_i	$h_{i,1}$	$h_{i,2}$	$h_{i,3}$	$h_{i,4}$	$h_{i,5}$
1	5.00	0.00	0.00	1.00	0.00	-1.00	10.00	2.00	0.00	0.00	0.00	0.00
2	10.00	2.00	0.00	0.00	0.00	0.00	10.00	0.00	2.00	0.00	0.00	0.00
3	10.00	0.00	-1.00	0.00	0.00	0.00	10.00	0.00	-1.00	2.00	0.00	0.00
4	8.00	0.00	0.00	2.00	0.00	-1.00	10.00	0.00	0.00	0.00	2.00	0.00
5	10.00	0.00	0.00	0.00	2.00	0.00	10.00	0.00	0.00	0.00	0.00	2.00

$$\tau^g_{Case1} = \begin{pmatrix} 0.0 & 0.0 & 0.0 & 0.0 & \mathbf{1.0} \\ 0.0 & \mathbf{1.0} & 0.0 & 0.0 & 0.0 \\ 0.0 & \mathbf{1.0} & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & \mathbf{1.0} \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix} \tau^h_{Case1} = \begin{pmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & \mathbf{1.0} & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix} \quad (6)$$

$$\tau^g_{Case2} = \begin{pmatrix} 0.0 & 0.0 & \mathbf{1.1} & 0.0 & 0.0 \\ \mathbf{1.2} & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & \mathbf{1.3} & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & \mathbf{1.0} \\ 0.0 & 0.0 & 0.0 & \mathbf{2.1} & 0.0 \end{pmatrix} \tau^h_{Case2} = \begin{pmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & \mathbf{1.0} & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix} \quad (7)$$

It can be observed that, proposed algorithm infer the first delayed network ('Case 1') accurately with high precision. On the other hand, the existing method [10], that does not have the knowledge of delayed interactions, misses inferring two of the delayed regulations. Moreover, using [10], correctly inferred parameters have poor precision. Due to the presence of delayed interactions in the GRN and hence the effect of the delayed interactions in the microarray data, REGARD [10] also failed to retrieve one instantaneous interaction as well. The proposed TDSS also reported excellent result while tested with 'Case 2'. The best and average case results of sensitivity and specificity, shown in Table 2, demonstrate that for both the cases with 10 and 20 samples, proposed method can infer all the delayed and instantaneous interactions correctly. Although, in the best case, TDSS could infer all the parameters correctly with $T = 10$ and $T = 20$ samples for both the networks (i.e., Case 1 and Case 2), the accuracy of average case results are improved with the increase of sample sizes (T), indicating a influence of number of samples in the data-set. It should be noted that the proposed TDSS not only inferred all the regulations correctly but also the with high precision, as shown in Table 3. The average errors (AE), which also reflects the preciseness of the regulatory weights, are shown in Fig. 1 indicating the comprehensive superiority of TDSS for reverse engineering time-delayed GRNs.

So far, the experiment has been carried out for synthetic network. We observe that, while the TDSS in [7] can infer all the S-System parameters (including delay matrices) within 1000 iterations, the TDSS used in this paper can infer the regulations only in less than half of the iterations than [7] when the delay matrices are known. The model and method can be easily applicable to real genetic

Table 2. Performance comparison for 5-gene synthetic network [1] with TDSS and REGARD [10]. Two different cases of delay are testing with S_n and S_p

	Case 1				Case 2			
	T=10		T=20		T=10		T=20	
	S_n	S_p	S_n	S_p	S_n	S_p	S_n	S_p
TDSS (Best)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
TDSS (Average)	0.91	0.94	1.00	1.00	0.90	0.81	0.92	0.94
	± 0.02	± 0.03	± 0.00	± 0.00	± 0.02	± 0.03	± 0.02	± 0.03
REGARD [10] (Best)	0.92	0.95	0.92	0.95	0.85	0.95	0.85	0.95
REGARD [10] (Average)	0.86	0.84	0.88	0.91	0.80	0.85	0.78	0.85
	± 0.02	± 0.03	± 0.01	± 0.04	± 0.02	± 0.01	± 0.01	±0.02

Table 3. Experimental results (best case) to infer 5-gene synthetic network [1] for two different cases: **"Case 1"** and **"Case 2"** for 10 data sets

Case 1												
	i	α_i	$g_{i,1}$	$g_{i,2}$	$g_{i,3}$	$g_{i,4}$	$g_{i,5}$	β_i	$h_{i,1}$	$h_{i,2}$	$h_{i,3}$	$h_{i,4}$	$h_{i,5}$
Proposed (TDSS)	1	4.32	0.00	0.00	1.04	0.00	-1.05	9.23	2.07	0.00	0.00	0.00	0.00
	2	9.48	2.05	0.00	0.00	0.00	0.00	9.43	0.00	2.06	0.00	0.00	0.00
	3	8.82	0.00	-0.99	0.00	0.00	0.00	8.87	0.00	-0.97	2.22	0.00	0.00
	4	7.12	0.00	0.00	2.13	0.00	-1.06	9.11	0.00	0.00	0.00	2.13	0.00
	5	9.24	0.00	0.00	0.00	2.09	0.00	9.25	0.00	0.00	0.00	0.00	2.12
REGARD [10]	1	2.56	-0.26	0.00	1.18	0.00	-1.19	7.96	2.79	0.00	0.00	0.00	0.00
	2	11.26	2.51	0.00	0.00	0.00	0.00	10.37	0.00	2.28	0.00	0.00	0.00
	3	4.21	0.00	-0.99	-0.61	0.00	0.00	6.37	1.30	0.00	0.00	0.00	0.00
	4	7.60	0.00	0.00	2.05	0.00	-1.09	9.84	0.00	0.00	0.00	1.86	0.00
	5	10.21	0.00	0.00	0.00	2.00	0.00	10.22	0.00	0.00	0.00	0.00	1.97
Case 2													
	i	α_i	$g_{i,1}$	$g_{i,2}$	$g_{i,3}$	$g_{i,4}$	$g_{i,5}$	β_i	$h_{i,1}$	$h_{i,2}$	$h_{i,3}$	$h_{i,4}$	$h_{i,5}$
Proposed (TDSS)	1	5.02	0.00	0.00	1.02	0.00	-0.97	10.23	2.04	0.00	0.00	0.00	0.00
	2	9.28	2.25	0.00	0.00	0.00	0.00	9.32	0.00	2.10	0.00	0.00	0.00
	3	9.02	0.00	-1.09	0.00	0.00	0.00	9.87	0.00	-0.99	2.12	0.00	0.00
	4	7.72	0.00	0.00	2.02	0.00	-1.05	9.51	0.00	0.00	0.00	2.01	0.00
	5	10.14	0.00	0.00	0.00	2.10	0.00	9.95	0.00	0.00	0.00	0.00	2.20
REGARD [10]	1	3.90	0.00	0.00	0.00	0.53	-0.83	9.32	1.15	0.00	0.00	0.00	0.00
	2	13.39	2.25	0.00	0.39	0.00	0.36	11.12	0.00	2.17	0.00	0.00	0.00
	3	19.74	0.00	-0.81	0.00	0.00	0.00	17.73	0.00	0.00	3.00	0.00	0.00
	4	13.22	0.00	0.00	2.14	0.00	-0.76	16.17	0.00	0.00	0.00	1.48	0.00
	5	20.00	0.00	0.00	0.00	1.91	0.00	16.81	0.00	0.00	0.00	0.00	2.34

network by performing a pre-processing stage to generate the delay metrics (τ^g and τ^h). The Pearson correlation coefficient, mutual information, conditional mutual information are few well-known and widely used statistical models to find the rate of interactions among variables (genes in this regard), although these methods will not provide the true delay metrics in most of the cases.

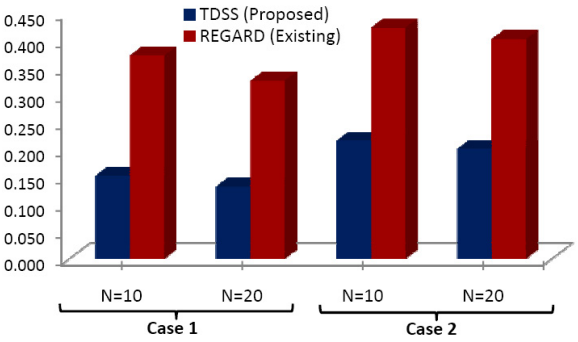


Fig. 1. Average errors reported by proposed and existing methods for two different delay networks in terms of number of samples in the time-series data

5 Conclusion

In this paper, we have evaluated the performance of newly proposed Time-delayed S-System model for various delayed networks with known delay. A new reconstruction algorithm, built upon the Trigonometric Differential Evolution (TDE), is designed based on the adaptive regulatory-genes cardinality based fitness function. A well-studied synthetic network, incorporating two delay scenarios, is considered for evaluation The performance is measured by the two widely accepted performance measures (i.e., Sensitivity and Specificity), where average errors are also plotted for both the cases. We have also investigated the effect of sample sizes in the reconstruction procedure for the same delayed networks. In all the cases, excellent performances are observed for TDSS over state-of-the-art algorithm. The focus of our current research is to extend the applications to medium and large-scale network for higher number of delayed interactions.

Acknowledgements. This work is supported in part by NICTA (National Information and Communication Technology Australia) research in Systems Biology flagship program.

References

1. Kikuchi, S., Tominaga, D., Arita, M., Takahashi, K., Tomita, M.: Dynamic modeling of genetic networks using genetic algorithm and S-System. Bioinformatics 19(5), 643–650 (2003)
2. Kimura, S., Ide, K., Kashihara, A., Kano, M., Hatakeyama, M., Masui, R., Nakagawa, N., Yokoyama, S., Kuramitsu, S., Konagaya, A.: Inference of S-System models of genetic networks using a cooperative coevolutionary algorithm. Bioinformatics 21(7), 1154–1163 (2005)

3. Chowdhury, A.R., Chetty, M.: An improved method to infer gene regulatory network using s-system. In: IEEE Congress on Evolutionary Computation (IEEE CEC), pp. 1012–1019 (2011)
4. Noman, N., Iba, H.: Inferring gene regulatory networks using differential evolution with local search heuristics. IEEE Transactions on Computational Biology and Bioinformatics 4, 634–647 (2007)
5. Savageau, M.: Biochemical Systems Analysis. A Study of Function and Design in Molecular Biology. Addison-Wesley Publishing Company, Massachusetts (1976)
6. Zou, M., Conzen, S.D.: A new dynamic bayesian network (dbn) approach for identifying gene regulatory networks from time course microarray data. Bioinformatics 21(1), 71–79 (2005)
7. Chowdhury, A.R., Chetty, M., Vinh, N.X.: Incorporating time-delays in S-System model for reverse engineering genetic networks. BMC Bioinformatics 14, 196 (2013)
8. Li, X., Rao, S., Jiang, W., Li, C.: Discovery of timedelayed gene regulatory networks based on temporal gene expression profiling. Bioinformatics 7, 7–26 (2006)
9. Xing, Z., Wu, D.: Modeling multiple time units delayed gene regulatory network usingdynamic bayesian network. In: International Conference on Data Mining, pp. 190–196 (2006)
10. Chowdhury, A.R., Chetty, M., Vinh, N.X.: Adaptive regulatory genes cardinality for reconstructing genetic networks. In: IEEE Congress on Evolutionary Computation (IEEE CEC), pp. 1–8 (2012)
11. Cho, D.Y., Cho, K.H., Zhang, B.T.: Identification of biochemical networks by s-tree based genetic programming. Bioinformatics 22, 1631–1640 (2006)
12. Kimura, S., Nakayama, S., Hatakeyama, M.: Genetic network inference as a series of discrimination tasks. Bioinformatics 25, 918–925 (2009)

Reverse Engineering Genetic Networks with Time-Delayed S-System Model and Pearson Correlation Coefficient

Ahsan Raja Chowdhury[1,2], Madhu Chetty[1,2], and Nguyen Xuan Vinh[1]

[1] Gippsland School of Information Technology, Monash University, Australia
[2] National Information and Communication Technology Australia (NICTA)
{ahsan.chowdhury,madhu.chetty}@monash.edu, vinh.nguyenx@gmail.com,
{ahsan.chowdhury,madhu.chetty}@nicta.com.au

Abstract. In almost all biological systems including genetic networks, the complex simultaneous interactions occurring amongst different organelles within a cell are both - instantaneous and time-delayed. Among the various modeling approaches, applied for inferring Gene Regulatory Network (GRN), recently proposed Time-delayed S-System Model (TDSS) is capable of simultaneously represent both the instantaneous and time-delayed interactions. While the delay parameters are incorporated in the S-System model to propose TDSS, this open a new challenge in GRN reconstruction. This paper proposes a systematic approach to fit in various level of knowledge in the delay parameters during the reverse engineering process. Further, we have approximated the delay parameters with well-known statistical measure Pearson correlation coefficient. Experimental studies have been carried out considering two widely used synthetic networks with various delays and real-life network of *Saccharomyces cerevisiae* called IRMA. The results clearly exhibit the influence of incorporating knowledge in the parameter learning process.

Keywords: Pearson correlation coefficient, Reverse Engineering, Time-delay.

1 Introduction

A Gene Regulatory Network (GRN) captures the mutual interactions among genes and helps in better understanding of the interactions at cellular level. The analysis of GRNs and metabolic pathways is carried out with various types of GRN models, e.g., Boolean Network, Petri Net, Bayesian Network (BN), Dynamic Bayesian Network (DBN) [1]. The S-System model, first proposed by Savageau [2], is a well-known formalism for modeling biochemical systems and was applied for GRN modeling in the late 90s. Despite its large number of parameters, the S-System formalism has an excellent ability to cope with nonlinearity and to infer a GRN accurately [2–6]. However, the applications of the S-System model was limited to very small networks due to the high computational complexity. Later, a decoupled version S-System greatly improved the

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 624–631, 2013.

overall performance of GRN reconstruction [3], but still the application is limited to medium scale network consists of 30-50 genes [5]. Other than our recently proposed TDSS [7], all the so far proposed method for GRN reconstruction with S-System model consider instantaneous interactions in the network, hence the delayed regulations are either missed or inferred with inappropriate weights. The TDSS [7] has included the time-delays in the regulatory interactions while inferring GRNs with S-System model and capable of simultaneously representing both instantaneous and time-delayed regulations. The proposed TDSS is capable of learning the delay parameters during the evolutionary optimization process, along with the other S-System parameters. As the new modeling parameters (delay) are included in the TDSS, it opens the further scope of optimization in the reverse engineering process. This paper presents a systematic approach to include knowledge in the delay parameters, excluding them from the learning process. In addition, we have proposed a new technique for generating the knowledge using modified Pearson correlation coefficient technique. The experiential results clearly demonstrates the effect of knowledge in the optimization.

2 Literature Review

2.1 The S-System Model

For a network of N genes, the S-System model is given by the following set of ordinary differential equations (ODEs):

$$\frac{dX_i}{dt} = \alpha_i \prod_{j=1}^{N} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_j^{h_{ij}}, \quad i = 1 \ldots N \tag{1}$$

Here, for any i^{th} gene, X_i is the expression level, $\{\alpha_i, \beta_i\, g_{ij}, h_{ij}\}$ are traditional S-System parameters that represent the network. To infer a GRN of N genes using the S-System model, $2N(N+1)$ parameters must be estimated, which are reduced to $2(N+1)$ by introducing the decoupled S-System [4, 5].

2.2 Time Delays in Biological Network

In the biological system, various genetic interactions occur amongst different genes concurrently. An interaction among two genes is said to be instantaneous if it occurs within single time-sample/timestamp (TS). On the other hand, if the interaction from the source gene starts in one TS and ends in a later TS it is known as time-delayed interaction. In the currently existing S-System model based reconstruction techniques the delayed interactions are missed or inferred with incorrect regulatory weights by the reconstruction method. However, our recently proposed TDSS [7] can reconstruct the delayed interactions, along with the instantaneous interactions, using the modified S-System model and improved inference method with compatible accuracies in the learning of system parameters.

3 The Proposed Framework for GRN Reconstruction

3.1 TDSS: The Model

It is already mentioned that, for a particular gene (gene-i) in a N-gene network, the S-System can represent $2N$ interactions with N interactions of production phase and remaining of degradation phase [2]. As a result, delay can occur in any of the $2N$ interactions. Hence we derive the time-delayed S-System equation accordingly:

$$\frac{dX_{i,t}}{dt} = \alpha_i \prod_{j=1}^{N} X_{j,(t-\tau^g_{i,j})}^{g_{ij}} - \beta_i \prod_{j=1}^{N} X_{j,(t-\tau^h_{i,j})}^{h_{ij}} \quad i = 1 \dots N, \quad t = 0 \dots T \qquad (2)$$

where, the delay parameters τ^g and τ^h are considered as two matrices of dimension $N * N$, defined as $\{0 \leq \{d^g_{i,j}, d^h_{i,j}\} \leq \tau_{max}\}$, $\forall_{i,j=1\dots N}$. Here, $d^g_{i,j}$ ($d^h_{i,j}$) indicate the delay unit for regulation on i from j in production (degradation) phase, and τ_{max} represents the maximum delay of the GRN. The new model requires both τ^g and τ^h as the input of the model or should be learned during the optimization along with other S-System parameters (α, β, g, h). For synthetic data, we have generated the time-series data from the network parameters by experimentally adding delays on predefined interactions in the network. Hence, we get the delay matrices directly from our estimation. However, delay matrices are not known in the same way for real genetic network and they should be constructed fully or partially by applying a suitable preprocessing technique, or should consider as parameters to learn during the optimization. We propose a heuristic pre-processing phase in the following section to obtain the delay matrices.

3.2 Constructing Delay Matrices with Modified Pearson Correlation Coefficient

We have used Pearson correlation coefficient (PCC) technique [8] to obtain the approximated lag values for the interaction based on our improved Pearson correlation coefficient. The conventional PCC is a measure of dependance (linear) between two variables X and Y (genes in our case), and returns a value between +1 and -1 inclusive. If we shift variable Y to s cell(s) and perform the PCC, it will return a value which indicates the mutual interaction of variable X on variable Y with s unit delay. However, due to the non-linear nature of genes' interactions and linearly testing property of the PCC, the precalculation of delay matrices yields the values that may have inappropriate meaning in terms of true regulatory network. We propose a heuristic algorithm to calculate more accurate matrices using PCC and then extract the approximate delay unit for each interaction. Lets consider that a GRN of N genes are sampled T times to produce a microarray data. Moreover, there are M datasets, which are produced from M initial conditions for each genes. In between two time-samples, we generate 9 more values using well-known Linear Spline Interpolation. At the

end, each data set has $10\times T$ samples which we will use to generate the delay matrices. After that, we apply the following improvement over standard PCC to obtain the probable delay for the interactions of the given network. We define the following operation as the standard PCC between gene-i (G_i) and gene-j (G_j) of k^{th} data set, shifting G_j expression by s cells to the right.

$$PCC^k(s)[i,j], s = 0 \ldots 10\tau_{max}, k = 1 \ldots M, i = 1 \ldots N, j = 1 \ldots N \quad (3)$$

Here, $s = 0$ implies conventional PCC operation, while other values of s define PCC for s unit delay between genes G_i and G_j. We set the maximum value of s to τ_{max} before computing the PCC based on some "prior knowledge" about the network. Every $PCC^k(s)[i,j]$ for a particular s value construct M matrices (for M data sets), each having $N \times N$ cells. According to the following equation, we obtain the average among M absolute values for each interaction:

$$PCC^0(s)[i,j] = Average(|PCC^1(s)[i,j]|, |PCC^2(s)[i,j]|, \ldots, |PCC^M(s)[i,j]|), \\ \forall i,j = 1, \ldots, N, \forall s = 0, \ldots, 10 \times \tau_{max} \quad (4)$$

Then we filter every cells of each matrix based on the threshold ζ (=0.9 for our experiment) and mark the cell by its corresponding s value:

$$PCC^0(s)[i,j] = \begin{cases} s \text{ if } PCC^0(s)[i,j] > \zeta \\ 0 \text{ otherwise} \end{cases}, \forall i,j = 1, \ldots, N, \forall s = 0, \ldots, 10 \times \tau_{max} \quad (5)$$

Finally, we obtain the PCC matrix for a microarray data by taking the maximum lag values among all $10 \times \tau_{max}+1$ values for every interaction:

$$PCC[i,j] = max\{PCC^0(0)[i,j], PCC^0(1)[i,j], \ldots, PCC^0(10 \times \tau_{max})[i,j]\} \\ \forall i,j = 1 \ldots N \quad (6)$$

This matrix is considered as the prior knowledge for the delayed regulations and used throughout the optimization in one separate experiment. Although, the proposed TDSS requires two delay matrices, i.e., τ^g and τ^h, we set the value of the matrix of Eqn. (6) to τ^g and consider 0 to all entries in τ^h. These values are initialized once to τ^g and τ^h, and remain fixed throughout the optimization.

3.3 Inference Method

In this research, we have used the recently proposed inference method 'REGARD' [9], which is a two phase optimizer. REGARD first initializes the population with a domain knowledge based 'Prediction Initialization Algorithm' that initializes 90% individuals with exactly I regulations, where I is the maximum in-degree of a genetic network, and remaining 10% individuals are initialized with zero regulation. Moreover, we have used the fitness function of REGARD [9] in this method to measure the goodness of a solution. During the optimization, where the procedures of Trigonometric Differential Evolution (TDE) are invoked

(i.e., Mutation, Crossover), a local search heuristic [5] is applied to iteratively check an individual by forcing low weighted values to zero. After certain generations or when the terminal condition is met, the first phase ends up with probable candidate solution. Then a two stage pruning algorithm (Multistage Refinement Algorithm) is invoked to further improve the learned parameters. It should be noted that, similar to REGARD [9] the max in-degree value for each gene is initialized to N, however, the min-degree in initialized to the total number of delayed regulations of the corresponding gene. The proposed TDSS is executed for 400 generations in each runs, while 5 runs are taken in total. In each run, the individual with minimum fitness value is reported as candidate solution.

4 Experimental Results and Discussions

We have segregated the experimentation in two sections: the first section demonstrate the performance of TDSS on two two synthetic networks, while the investigation results on real-life IRMA network are shown in the following section. For the first section, we have incorporated knowledge of various levels (i.e., 0%, 50%, 75%, 100%) for the delay matrices, as the delays for each regulations are known. Further, we have shown the results of TDSS by including the knowledge about the delay calculated using the modified PCC technique. Since, the delayed regulations are not known as prior knowledge for the real-life networks, we have applied the information extracted from our modified PCC as the value for delay parameters.

4.1 Synthetic Networks

The performance of the new model TDSS is studied by investigating the 5-gene synthetic network of [3]. For comparison, we have implemented the same experimental setup reported in [9]. Two well-known performance measures, i.e., Sensitivity (S_n) and Specificity (S_p), are used to demonstrate the performance of TDSS. We have considered two different cases of the 5-gene synthetic network, can be found as Conf-2 and Conf-3, respectively in [7]. It is already mentioned that, both the delay matrices (τ^g and τ^h) should be given as the input to the optimization method, which is known to verify the synthetic networks. Considering this known matrices are knowledge, we have investigated the efficacy of the proposed method for different levels of knowledge, and results are shown in Table 1. Case 1 has single TS delay on 5 regulations, while Case 2 has four interactions with fractional TS values, and two more interactions with integer TS. It can be observed that, proposed algorithm infer the delayed network ('Case 1') accurately with high precision. Due to the presence of delayed interactions in the GRN and hence the effect of the delayed interactions in the microarray data, proposed TDSS failed to retrieve one instantaneous regulation up to 50% of knowledge was provided about the delay matrices. However, TDSS was successful to infer all the regulations correctly when 75% of knowledge was incorporated,

while the entire network was inferred correctly with the full knowledge. Due to the fractional delays in 'Case 2', TDSS failed to infer all the regulations without incorporating 100% knowledge. We also test the performance of TDSS when the given knowledge is constructed from the modified PCC. Although not optimal, the performance of TDSS for both the Cases are satisfactory and indicate the effect of modified PCC in creating delay matrices. The proposed TDSS is also

Table 1. Experimental result for infer 5-gene Synthetic Network [3] with TDSS for two different cases: "Case 1" and "Case 2"

		Given Knowledge in Delay Matrices (τ^g and τ^h)									
		0%		50%		75%		100%		Using PCC%	
		S_n	S_p	S_n	S_p	S_n	S_p	S_n	S_p	S_n	S_p
Case 1	Best Case	0.92	0.89	0.92	0.94	1.00	0.97	1.00	1.00	0.92	0.90
	Average Case	0.86 ±0.01	0.84 ±0.01	0.89 ±0.01	0.90 ±0.01	0.93 ±0.02	0.94 ±0.02	0.95 ±0.01	0.98 ±0.01	0.90 ±0.02	0.87 ±0.03
Case 2	Best Case	0.86	0.70	0.91	0.84	0.94	0.90	1.00	1.00	0.90	0.87
	Average Case	0.84 ±0.02	0.65 ±0.03	0.86 ±0.02	0.75 ±0.03	0.86 ±0.02	0.80 ±0.03	0.94 ±0.02	0.84 ±0.05	0.87 ±0.02	0.84 ±0.03

evaluated with a 20-gene synthetic network [5] with a delay configuration shown as Conf-5 in [7]. The results, shown in Fig. 1, demonstrate that with the 100% knowledge given as input the proposed TDSS can infer the entire network within considered 400 iterations. However, the performance of TDSS for 0% knowledge which indicates the clear influence of delayed regulations in the microarray data. Moreover, when the knowledge is constructed from modified PCC the result is again not very satisfactory, which implies the lack of accuracy for PCC in detecting the non-linear dependency between regulations in a delayed network.

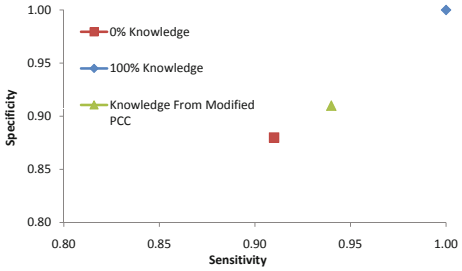


Fig. 1. ROC graph of TDSS for inferring 20-gene delayed network with various levels of given knowledge in delay matrices.

4.2 Real-Network of IRMA

The proposed technique is next applied to a real-life biological network of *Saccharomyces cerevisiae* (*yeast*) called IRMA [10]. The network is composed of five genes ($CBF1$, $GAL4$, $SWI5$, $GAL80$, $ASH1$), regulating each other. There are two sets of gene expression profiles, namely Switch ON and Switch OFF data having 16 and 21 time series data points, respectively. In our experiment, we have used the Switch ON data set which consisting of 8 regulations in the original network. The evaluation is performed in two ways: no knowledge were given as input in the first case, while in the later case, delay parameters were given as input which is obtained by modified Pearson correlation coefficient technique (discussed in Section 3.2). The experimental result, shown in Table 2, highlights that the proposed TDSS with delayed matrix performs better than all the existing methods in literature in terms of Sensitivity. Although, the Specificity is slightly inferior, still comparable to the existing state-of-the-art methods.

Table 2. Performance Comparison with IRMA real network in terms of Sensitivity (S_n) and Specificity (S_p) among existing and proposed TDSS

	TDSS				TDARACNE [11]	NIR & TSNI [12]	BANJO [13]
	without PCC		with PCC				
	(Best)	(Average±StdDev)	(Best)	(Average±StdDev)	(Best)	(Best)	(Best)
S_n	0.69	0.65 ±0.04	0.76	0.70±0.03	0.63	0.50	0.25
S_p	0.83	0.80±0.02	0.86	0.82±0.03	0.88	0.94	0.76

5 Conclusion

In this paper, we have evaluated the performance of Time-delayed S-System (TDSS) model for various Synthetic and Real-networks. While the delay parameters are included in the S-System equations to form TDSS model, the optimization in the delay parameters opens a new field of research. In contrast to learning the delay parameters, we have proposed an heuristic approach to estimate those parameters using modified Pearson correlation coefficient. The experiments have been carried out by including various levels of knowledge (% of total delayed regulations) for different delayed networks. We observe a gradual improvement in the learning process with the increase of knowledge level in the delay parameter. Investigations carried on both the synthetic networks and real networks show that our approach outperforms (in terms of sensitivity and specificity) the existing methods used for comparison in this paper.

Acknowledgements. This work is supported in part by NICTA (National Information and Communication Technology Australia) research in Systems Biology flagship program.

References

1. de Jong, H.: Modeling and simulation of genetic regulatory systems: A literature review. Journal of Computational Biology 9(1), 67–103 (2002)
2. Savageau, M.: Biochemical Systems Analysis. A Study of Function and Design in Molecular Biology. Addison-Wesley Publishing Company, Massachusetts (1976)
3. Kikuchi, S., Tominaga, D., Arita, M., Takahashi, K., Tomita, M.: Dynamic modeling of genetic networks using genetic algorithm and s-system. Bioinformatics 19(5), 643–650 (2003)
4. Kimura, S., Ide, K., Kashihara, A., Kano, M., Hatakeyama, M., Masui, R., Nakagawa, N., Yokoyama, S., Kuramitsu, S., Konagaya, A.: Inference of s-system models of genetic networks using a cooperative coevolutionary algorithm. Bioinformatics 21(7), 1154–1163 (2005)
5. Noman, N., Iba, H.: Inferring gene regulatory networks using differential evolution with local search heuristics. IEEE Transactions on Computational Biology and Bioinformatics 4, 634–647 (2007)
6. Chowdhury, A.R., Chetty, M.: An improved method to infer gene regulatory network using s-system. In: IEEE Congress on Evolutionary Computation (IEEE CEC), pp. 1012–1019 (2011)
7. Chowdhury, A.R., Chetty, M., Vinh, N.X.: Incorporating time-delays in S-System model for reverse engineering genetic networks. BMC Bioinformatics 14, 196 (2013)
8. Stigler, S.M.: Francis galton's account of the invention of correlation. Statistical Science 4(2), 73–79 (1989)
9. Chowdhury, A.R., Chetty, M., Vinh, N.X.: Adaptive regulatory genes cardinality for reconstructing genetic networks. In: IEEE Congress on Evolutionary Computation (IEEE CEC), pp. 1–8 (2012)
10. Cantone, I., Marucci, L., Iorio, F., Ricci, M.A., Belcastro, V., Bansal, M., Santini, S., di Bernardo, M., di Bernardo, D., Cosma, M.P.: A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches. Cell 137, 172–181 (2009)
11. Zoppoli, P., Morganella, S., Ceccarelli, M.: Timedelay-aracne: Reverse engineering of gene networks from time-course data by an information theoretic approach. BMC Bioinformatics 11(1), 154 (2010)
12. Della, G.G., Bansal, M., Ambesi-Impiombato, A., Antonini, D., Missero, C., di Bernardo, D.: Direct targets of the trp63 transcription factor revealed by a combination of gene expression profiling and reverse engineering. Genome Research 18, 939–948 (2008)
13. Yu, J., Smith, V.A., Wang, P.P., Hartemink, A.J., Jarvis, E.D.: Advances to bayesian network inference for generating causal networks from observational biological data. Bioinformatics 20, 3594–3603 (2004)

EEG-Based Age and Gender Recognition Using Tensor Decomposition and Speech Features

Phuoc Nguyen[1], Dat Tran[1], Tan Vo[1], Xu Huang[1], Wanli Ma[1,2], and Dinh Phung[1]

[1] Faculty of Education, Science, Technology & Mathematics
University of Canberra, ACT 2601, Australia
[2] Department of Computer Science, University of Houston Downtown, USA
{phuoc.nguyen,dat.tran,tan.vo,xu.huang,wanli.ma}@canberra.edu.au

Abstract. Extracting age and gender information from EEG data has not been investigated. This information is useful in building automatic systems that can classify a person into gender or age groups based on EEG characteristics of that person, index EEG data for searching, identify or verify a person, and improve performance of brain-computer interface systems. In this paper, we propose a framework based on PARAFAC and SVM that can automatically classify age and gender using EEG data. We also propose a method using N-PLS and SVM to improve the classification rate. Experimental results for the proposed method are presented.

Keywords: Electroencephalogram (EEG), age and gender Recognition, Speech Features, parallel factors (PARAFAC), multilinear partial least squares (N-PLS), support vector machine (SVM).

1 Introduction

In clinical psychophysiology, the effects of age and gender on electroencephalogram (EEG) signal have been investigated [1], [2]. Topographic differences in EEG maturation have been found. The differences in the EEGs of normal children and children with attention-deficit/hyperactivity disorder was also reported [1]. Particularly, there is a shrinkage of large neurons and an increasing the number of small neurons and glia [3]. The reason for these changes could be the decrease in cerebral blood flow [4]. When age increases the low frequencies of EEG seem to be replaced by faster waves. In children, the delta and theta waves are dominant before the age of 4 years then decreasing. Meanwhile the alpha and beta waves increase throughout childhood and the total amplitude decreased with age during childhood [5]. In the elderly, the reduction of the alpha frequency causes a greater anterior spread to frontal regions and reduces the alpha wave blocking response and reactivity. Increment in beta wave activities has been reported and considered as an early indication of intellectual loss [3]. Bioelectrical correlates of face gender recognition was investigated in [6], where visual stimulation was performed with 64 pictures of human faces. However extracting age and gender information from EEG data has not been addressed as

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 632–639, 2013.

a problem in knowledge discovery and data mining area. This information is useful in building automatic systems that can classify a person in to gender or age groups based on EEG characteristics of that person, index EEG data for searching, identify or verify a person based on his/her EEG data, and improve performance of brain-computer interface systems.

In neuroscience, principal component analysis (PCA) and independent component analysis (ICA) have been mainly utilised for decomposing multi-channel EEG signals. These methods, however, require two-dimensional representation of the data and have disadvantages when high-dimensional representation is needed. Recently, multi-way analysis or tensor decomposition has gained many interests due to the multi-dimensional nature of neuroscience data. Tensor decomposition has been used to model EEG signal as space-time-frequency arrays. Specifically, PARAFAC was used to decompose wavelet transformed event-related EEG consisting of either channel-frequency-time 3-way arrays, [7] and [8], or channel-frequency-time-subject-condition 5-way arrays, [7].

In [9], multiway partial least squares (N-PLS) was used to analyse concurrent EEG/fMRI data. The EEG (independent variable) was decomposed as spatial-spectral-temporal atoms and the fMRI (dependent variable) as spatial-temporal atoms with the constraints to maximize the covariance between temporal signatures of the EEG and fMRI. In [10], canonical decomposition (CANDECOMP) was used to detect the seizure onset zone from the potential distribution of the ictal activity from EEG. The author showed that only one atom is related to the seizure activity. In [11], PARAFAC was employed to detect neonatal seizure localization by using a sum of rank-1 components to extract oscillatory seizure activity and spike train activity. In [12], PARAFAC was used to model the epilepsy seizure structure, localize a seizure origin and extracting artifacts. An epilepsy tensor was constructed with three modes time-scales-electrodes using wavelet analysis of multi-channel ictal EEG. In [13], PARAFAC was combined with SVM to classify left and right index imagery movements. The spatial-temporal-spectral characteristics of the single trial EEG signal were decomposed into two distinct factors and SVM was utilised to classify these factors. More reviews of tensor decompositions and its applications can be found in [14].

We propose in this paper a framework based on PARAFAC and SVM to automatically classify age and gender from EEG signal. Features extracted from each EEG channel consist of popular EEG features and the speech processing-based features. The latter feature extraction method was also used in [15] and [16] to detect neonatal seizures. After extracting features, we used these features to build a third-order tensor with the following modes: time epochs, features and electrodes. Then we use PARAFAC to decompose the tensor into sum of rank-one tensors. These tensors were then used to extract features and sent to a machine learning method such as SVM to build age and gender models for that person. Additionally, in order to improve the recognition rate we propose a method that utilise the N-PLS regression model to extract the components that can fit the tensor and predict the class labels at the same time. These components were used to extract features for the SVM classifier. Experimental

results show that our propose system can recognise age and gender from EEG signal and using N-PLS and SVM can improve the recognition rate.

2 EEG Database

The Australian EEG Database used in this research consists of EEG recordings of 40 patients at the John Hunter Hospital [17], near University of Newcastle, over an 11-year period. There are 20 male and 20 female and their ages are between 19 and 69. The EEGs were recorded using 23 electrodes followed the standard International System 10-20 electrode placements. The recordings were sampled at 167 Hz for about 20 minutes in the resting state with eyes open and eyes closed. The recordings were split equally into a training set to train age and gender models and a test set to evaluate the performance of the age and gender classification system.

3 Feature Extraction

As described in the Introduction section, we extracted features for the EEG data and saved them in 2 sets. The first set contains the popular EEG features (relative powers in different frequency bands, auto regressive and Hjorth parameters) and the second set contains the speech features (MFCC, Log filter-bank powers and Line spectral pairs).

Popular EEG Features: The spectral power in 2 Hz frequency bins from 1 to 30 Hz was computed for each channel. The central frequency of each bin was an integer. The relative power, which is the power in a specific frequency divided by the total power in all frequency bins from 1 to 30 Hz, together with the total power were also used. In addition, 11 AR coefficients of the 11th-order AR model and 3 Hjorth parameters (activity, mobility and complexity) were extracted for each electrode site.

Speech Features: We used the open-source Emotion and Affect Recognition toolkit's feature extraction backend openSMILE [18] for extracting speech features. Each channel is considered as an EEG signal and is extracted speech features. The features include 10 Mel-frequency cepstral coefficients (MFCC) calculated from the discrete cosine transform (DCT) the log filterbank powers, 15 log filter-bank powers (LFBP) computed from the corresponding filter-banks [18] and 8 line spectral pairs (LSP) coefficients. LSP have smaller sensitivity to quantization noise than linear prediction coefficients (LPC) hence are useful in speech coding and speaker recognition [19].

4 Parallel Factor Analysis (PARAFAC)

The PARAFAC model with R components of a three-way array $\underline{\mathbf{X}}(I \times J \times K)$, with elements x_{ijk} can be written as a trilinear model [20]:

$$x_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} + e_{ijk} \tag{1}$$

or using the tensor products notation:

$$\underline{\mathbf{X}} = \sum_{r=1}^{R} \mathbf{a}_r \Delta \mathbf{b}_r \Delta \mathbf{c}_r + \underline{\mathbf{E}} \tag{2}$$

where $\mathbf{a}_r$, $\mathbf{b}_r$, $\mathbf{c}_r$ with elements a_{ir}, b_{jr}, c_{kr} respectively are the rth columns of the component matrices $\mathbf{A}(I \times R)$, $\mathbf{B}(J \times R)$ and $\mathbf{C}(K \times R)$ respectively and $\underline{\mathbf{E}}(I \times J \times K)$ is a three-way array containing the residuals with elements e_{ijk}.

The PARAFAC decomposition was performed using orthogonal loadings. After the decomposition, the matrices $\mathbf{B}$ and $\mathbf{C}$ were used to project the tensor to the factor subspace. The resulting tensor was then matricized to make feature vectors for feeding to an SVM classifier.

The PARAFAC decomposition can decompose the tensor into components that best explain the variance of the tensor. However these components may be good in one way of labeling data but may not be good in other ways of labeling them. Some possible ways to label EEG data can be by age, gender, epileptic seizure and motor imagery information. Especially when various features of EEG signal are extracted from many electrodes and some features and electrode sites are better in explaining one kind of labeling than others. This problem motivates us to consider utilising the N-PLS regression model to extract the components that can at the same time fit the tensor and predict the class labels. These components were then used to extract features for the SVM classifier.

5 Multi-linear Partial Less Square (N-PLS) Regression

The N-PLS regression or three-way regression problem find a connection between a three-way array $\underline{\mathbf{X}}$ and a vector $\mathbf{y}$ through trying to predict $\mathbf{y}$ and decomposes $\underline{\mathbf{X}}$ in a PARAFAC-like model at the same time. It can be written [20]:

$$\mathbf{T} = \mathbf{XV} \tag{3}$$

$$\mathbf{X} = \mathbf{TW}' + \mathbf{E}_{\mathbf{X}}; \quad \mathbf{W} = [\mathbf{w}_1^K \otimes \mathbf{w}_1^J \| ... \| \mathbf{w}_R^K \otimes \mathbf{w}_R^J] \tag{4}$$

$$\mathbf{y} = \mathbf{T}\mathbf{b}_R + \mathbf{e}_{\mathbf{y}} \tag{5}$$

$$\underset{\mathbf{w}_r^J, \mathbf{w}_r^K}{max}\ cov(\mathbf{t}_r, \mathbf{y}^{r-1}); r = 1, ..., R \tag{6}$$

where $\mathbf{T}$ is a matrix of component vectors $\mathbf{T} = [\mathbf{t}_1, ..., \mathbf{t}_R]$, $\mathbf{X}$ is a properly unfolded two-way form of three-way array $\underline{\mathbf{X}}$, $\mathbf{b}_R$ is the regression coefficient, $\mathbf{W}$

and $\mathbf{V}$ are matrices of weighing coefficients, where $\mathbf{W}$ is not orthogonal and $\mathbf{V}$ can be written in terms of $\mathbf{w}_1 = (\mathbf{w}_1^K \otimes \mathbf{w}_1^J)$ to $\mathbf{w}_R = (\mathbf{w}_R^K \otimes \mathbf{w}_R^J)$:

$$\mathbf{V} = [\mathbf{w}_1 \;\; (\mathbf{I} - \mathbf{w}_1\mathbf{w}_1')\mathbf{w}_2 \;\; .. \;\; (\mathbf{I} - \mathbf{w}_1\mathbf{w}_1')(\mathbf{I} - \mathbf{w}_2\mathbf{w}_2')...(\mathbf{I} - \mathbf{w}_{R-1}\mathbf{w}_{R-1}')\mathbf{w}_R] \quad (7)$$

In test phase a component matrix $\mathbf{T}_t = \mathbf{X}_t\mathbf{V}$ was calculated from equation (3) and used as feature vectors for test data $\mathbf{X}_t$. The matrix $\mathbf{V}$ can be considered as feature selection matrix.

6 Experiments

6.1 Experiment Setup

The recordings in training and test phases were splitted into 15s epochs. The 8 channels F3, F4, C3, C4, P3, P4, O1 and O2 in the frontal, central, parietal and occipital sites were used. The channel signals in each epoch were used to extract features and these features were merged together to make a single feature vector for each epoch. The selected features are the popular EEG features (relative powers in different frequency bands, auto regressive and Hjorth parameters) or the speech features (MFCC, Log filter-bank powers and Line spectral pairs) presented above.

All feature vectors were first scaled and centered. They are then labeled for 3 age groups and 2 gender groups resulting to 6 classes which are young female, young male, middle age female, middle age male, elderly female and elderly male. The young age range is 19-34, middle age range is 35-54 and elderly is 55-69.

A third-order tensor was constructed in each of the training set and test set from the above features with modes time epochs, features and electrodes. Then the PARAFAC decomposition or N-PLS regression were performed on the training tensor to extract factors and loadings.

SVM was trained using Gaussian RBF kernel function $K(x_i, x_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2}$. Experiments were conducted using 10-fold cross validation on training set to find the best parameters and age and gender SVM models and these models were used to predict age and gender in test set. SVM parameter was search in the grid $\{(C, \gamma)|C = \{2^{-3}, ..., 2^{15}\}$ and $\gamma = \{2^{-15}, ..., 2^3\}\}$

6.2 Results

Figure 1 shows the recognition rate of the two method in training phase and test phase using different number of tensor factors in PARAFAC and N-PLS model. It can be seen from the figure that the number of tensor factors should be greater than 6 for the robust age and gender recognition, the performances are low when less than 4 factors are used.

Tables 1 and 2 show the confusion matrices of age and gender classification using PARAFAC-SVM and N-PLS SVM respectively. These matrices were calculated from the confusion matrix of 6-class classification experiment in the test

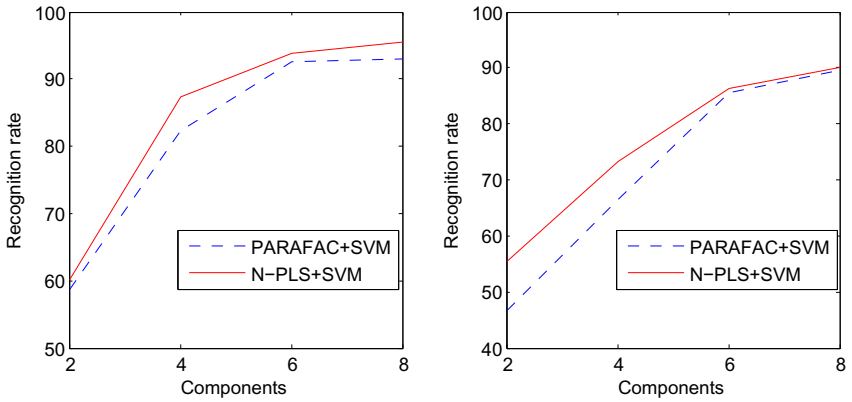


Fig. 1. Accuracy versus number of factors in EEG-based age and gender recognition. Left figure is training phase and right figure is testing phase

Table 1. Confusion matrix of age classification in test phase using 8-factor N-PLS+SVM (A) (with classification rate of 92.5%), and using 8-factor PARAFAC (B)(with classification rate of 91.9%

	Classified as →	Young	Middle	Elderly
(A)	Young	494	6	17
	Middle	29	599	23
	Elderly	4	40	374
	Classified as →	Young	Middle	Elderly
(B)	Young	481	12	24
	Middle	14	592	45
	Elderly	9	24	385

Table 2. Confusion matrix of gender classification in test phase using 8-factor N-PLS+SVM (A) (with classification rate of 93.8%), and using 8-factor PARAFAC (B)(with classification rate of 93.4%)

	Classified as →	Female	Male
(A)	Female	718	38
	Male	61	769
	Classified as →	Female	Male
(B)	Female	715	41
	Male	63	767

phase by summing the predictions over the desired classes. The tables show better performance in age and gender classification using the N-PLS SVM method.

Table 3 showed the top 10 factor loading coefficients getting from the first factor loading vector $\mathbf{w}_1^J$ averaged across the channel dimension in equation (4). It can be seen that the log filter band power, line spectral pair and cepstral coefficients have the most loading weights for the first factor

Table 3. Top 10 factor loading coefficients getting from the first factor loading vector $\mathbf{w}_1^J$ averaged across the channel dimension

#	Loading coefficients	Features
1	0.30538	logMelFreqBand[5]
2	0.30268	logMelFreqBand[10]
3	0.26764	lspFreq[7]
4	0.25214	logMelFreqBand[14]
5	0.24784	mfcc[4]
6	0.23394	logMelFreqBand[1]
7	0.22775	lspFreq[3]
8	0.22163	mfcc[8]
9	0.16134	logMelFreqBand[13]
10	0.16121	mfcc[8]

7 Conclusion

We have shown that age and gender information can be extracted from EEG signal and can be exploited for future applications. Feature extraction methods for speech signals which are normally used for age, gender and speaker recognition can also be employed for age and gender classification using EEG data. We have also demonstrated that tensor decomposition can be used to decompose the EEG and the speech features tensor. In addition the N-PLS regression can be used to improve the performance by extracting the components that can fit the tensor and predict the class labels at the same time.

References

1. Clarke, A.R., Barry, R.J., McCarthy, R., Selikowitz, M.: Age and sex effects in the eeg: development of the normal child. Clinical Neurophysiology 112(5), 806–814 (2001)
2. Carrier, J., Land, S., Buysse, D.J., Kupfer, D.J., Monk, T.H.: The effects of age and gender on sleep eeg power spectral density in the middle years of life (ages 20–60 years old). Psychophysiology 38(2), 232–242 (2001)
3. Van Sweden, B., Wauquier, A., Niedermeyer, E.: Normal aging and transient cognitive disorders in the elderly. Electroencephalography: Basic Principles, Clinical Applications and Related Fields 4, 340–348 (1999)
4. Sanei, S., Chambers, J.A.: EEG signal processing. Wiley-Interscience (2008)
5. Gasser, T., Verleger, R., Bächer, P., Sroka, L.: Development of the eeg of school-age children and adolescents. i. analysis of band power. Electroencephalography and Clinical Neurophysiology 69(2), 91–99 (1988)
6. Borghetti, D., Cellerino, A., Logi, F., Murri, L., Sartucci, F.: Neurophysiological correlates of face gender recognition. Journal of Endocrinological Investigation 26(suppl. 3), 148 (2003)
7. Mørup, M., Hansen, L.K., Herrmann, C.S., Parnas, J., Arnfred, S.M.: Parallel factor analysis as an exploratory tool for wavelet transformed event-related eeg. NeuroImage 29(3), 938–947 (2006)

8. Weis, M., Romer, F., Haardt, M., Jannek, D., Husar, P.: Multi-dimensional space-time-frequency component analysis of event related eeg data using closed-form parafac. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2009, pp. 349–352. IEEE (2009)
9. Martınez-Montes, E., Valdés-Sosa, P.A., Miwakeichi, F., Goldman, R.I., Cohen, M.S.: Concurrent eeg/fmri analysis by multiway partial least squares. NeuroImage 22(3), 1023–1034 (2004)
10. De Vos, M., Vergult, A., De Lathauwer, L., De Clercq, W., Van Huffel, S., Dupont, P., Palmini, A., Van Paesschen, W.: Canonical decomposition of ictal scalp eeg reliably detects the seizure onset zone. NeuroImage 37(3), 844–854 (2007)
11. Deburchgraeve, W., Cherian, P.J., De Vos, M., Swarte, R.M., Blok, J.H., Visser, G.H., Govaert, P., Van Huffel, S.: Neonatal seizure localization using parafac decomposition. Clinical Neurophysiology 120(10), 1787–1796 (2009)
12. Acar, E., Aykut-Bingol, C., Bingol, H., Bro, R., Yener, B.: Multiway analysis of epilepsy tensors. Bioinformatics 23(13), i10–i18 (2007)
13. Nazarpour, K., Sanei, S., Shoker, L., Chambers, J.A.: Parallel space-time-frequency decomposition of eeg signals for brain computer interfacing. In: Proc. EUSIPCO 2006 (2006)
14. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Review 51(3), 455–500 (2009)
15. Temko, A., Boylan, G., Marnane, W., Lightbody, G.: Speech recognition features for eeg signal description in detection of neonatal seizures. In: 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 3281–3284. IEEE (2010)
16. Temko, A., Nadeu, C., Marnane, W., Boylan, G.B., Lightbody, G.: Eeg signal description with spectral-envelope-based speech recognition features for detection of neonatal seizures. IEEE Transactions on Information Technology in Biomedicine 15(6), 839–847 (2011)
17. Hunter, M., Smith, R., Hyslop, W., Rosso, O.A., Gerlach, R., Rostas, J.A.P., Williams, D.B., Henskens, F.: The australian eeg database. Clinical EEG and Neuroscience 36(2), 76–81 (2005)
18. Eyben, F., Wöllmer, M., Schuller, B.: Opensmile: the munich versatile and fast open-source audio feature extractor. In: Proceedings of the International Conference on Multimedia, pp. 1459–1462. ACM (2010)
19. Campbell Jr., J.P.: Speaker recognition: A tutorial. Proceedings of the IEEE 85(9), 1437–1462 (1997)
20. Smilde, A., Bro, R., Geladi, P.: Multi-way analysis: applications in the chemical sciences. Wiley. com (2005)

A Hybrid Cancer Prognosis System Based on Semi-Supervised Learning and Decision Trees

Yonghyun Nam and Hyunjung Shin*

Department of Industrial Engineering, Ajou University,
Wonchun-dong, Yeongtong-gu, Suwon 443-749, South Korea
{namyh123,shin}@ajou.ac.kr

Abstract. Diverse machine learning models have applied to cancer survivability prediction. But most of them tend to report only the performance of the model. However, in order to help medical specialists to establish a treatment plan by using machine learning models, it is more pragmatic to elucidate which variables (markers) have most significantly influenced to the resulting outcome of cancer. This motivated us to propose a hybrid approach of two machine learning models, semi-supervised learning co-training and decision trees. The former performs prediction for cancer survivability, and the latter post-processes the results mainly focusing on which variables are more highly ranked. The proposed method was tested on the breast cancer survivability problem based on the surveillance, epidemiology, and end results database for breast cancer (SEER).

1 Introduction

Although breast cancer is the second most lethal cancer in women, the mortality rates have declined by about 25% since 1990, due to early detection, better treatment options, and particularly increased accuracy in cancer prognosis [1, 2]. Cancer prognosis includes cancer susceptibility, cancer recurrence, and cancer survivability [3-5]. Diverse predictive models from machine learning or data mining have employed to perform predictions on cancer survivability which stands for the problem of whether a patient is to be or not to be a survivor after 1,825 days (5 years) from the date of cancer diagnosis.

In [6], the authors conducted a wide ranging investigation of different machine learning methods, discussing issues related to the types of data incorporated and the performance of these techniques in breast cancer prognosis. The authors of [5] used two popular data mining algorithms, artificial neural networks (ANN) and decision trees (DT), together with a common statistical method, logistic regression, to develop prediction models for breast cancer survivability. In [4], a hybrid prognostic scheme based on weighted fuzzy decision trees is proposed, which has shown to be an effective alternative to independent use of crisp classifiers. In [7], support vector machines

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 640–648, 2013.

(SVM) based classification was carried out concerning both the prognosis and diagnosis problems of breast cancer. From the comparison results with ANN and Bayesian method, they demonstrates the superiority of SVM in terms of sensitivity, specificity and accuracy.

In prediction of survival of breast cancer patients, the performance of the established machine learning models including ANN, SVM, semi-supervised learning (SSL), Bayesian Methods, etc., has been often compared and the winner model is renewed paper by paper [7-10]. While such studies have been devoted to enhancement of the predictive power or accuracy of the predictive model, interpretability of the predicted results has received less attention. Most of them are like a black-box module only producing the prediction results and accuracy as a measure of performance for comparison. In other words, it is difficult to know what happened during prediction and how we obtained the results: for instance, the question like 'which factors(variables) are most significantly contributed to survival/death classification?' is usually veiled. In practice, however, the answer benefits for medical practitioners and patients in many ways. By knowing the significant factors, we can make a proper choice of therapy, which may elevate the likelihood of successful treatments. At the same time, redundant or unimportant factors for breast cancer can be ruled out from then on, which will lead to reduction in time and cost during data collection and during treatment as well. Among the representatives in machine learning models, a DT is a model equipped with reasonably good general ability and interpretability [10, 12, 16-18]. However, it would occur that its performance does not reach to those of the up-to-date models, i.e., SVM, SSL, Bayesian Methods [13, 14]. To investigate the predicted results further, one may not want to simply give up using the winner model.

To circumvent the dilemma, we suggest a hybrid approach of two machine learning models, SSL Co-training [8] and DT [17]. SSL Co-training generates the predicted output for cancer survivability. The model generates pseudo-labels by co-training multiple SSL member models, which assign them to unlabeled data before treating them as if they were labeled. As the labeled data increase, the predictive performance of the ordinary SSL increases. The algorithm realizes the tenet of 'the more labeled data, the better prediction' which would be applied to most machine learning algorithms. After prediction, DT post-processes the results in order to provide variable importance: which variables are more highly or less significantly ranked when describing the results of the prediction. The proposed method is validated on the surveillance, epidemiology, and end results cancer incidence database (SEER), which is known as the most comprehensive source of information on cancer incidence and survival [20]. Performance comparison of the proposed method with the latest machine learning models is provided with clinical implications on the results.

2 Proposed Method

The proposed model employs two models: SSL Co-training and DT. SSL Co-training generates the predicted labels for patient samples on whether the patient will be survived or not [8]. After then, DT post-processes the prediction results by using decision trees [16-18]. It profiles the reasons which variables are most determinant in identifying survived/dead patients, which translates as variable importance.

2.1 Prediction with Semi-Supervised Learning Co-training

Recently, many machine learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce considerable improvement in learning accuracy. And it is "paradigmed" as semi-supervised learning. SSL exploits the knowledge of the input structure from unlabeled data and at the same time utilizes the label information provided by labeled data [13, 14]. SSL may be a good candidate to use a predictive model for cancer survivability, particularly when the available dataset for model learning has an abundance of unlabeled patient cases but a lack of labeled ones. Like many other machine learning algorithms, however, the availability of more labeled data leads to better performance. This motivated our previous work, SSL Co-training, which is designed to generate "pseudo-labels" and increases the performance of SSL. The model is based on graph-based SSL [13, 14, 19].

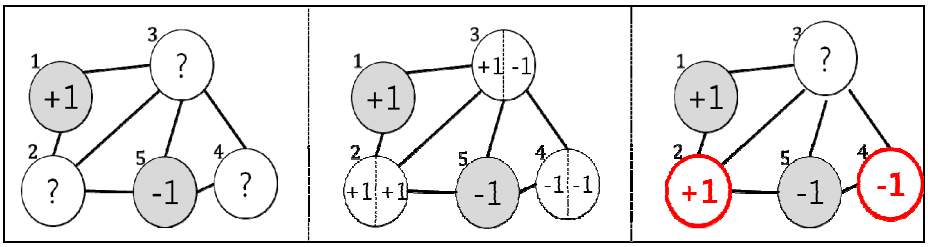
In graph-based SSL, a weighted graph is constructed where the nodes represent the labeled and unlabeled data points while the edges reflect the similarity between data points. Figure 1(a) depicts a graph with two labeled and three unlabeled data points. Given $n(= n_l + n_u)$ data points, the labeled nodes are set to $y_l \in \{-1, +1\}$, while the unlabeled nodes are set to zero ($y_u = 0$). The edge between the two nodes is usually measured by the Gaussian function, and the value of the similarity is represented by a matrix $W = \{w_{ij}\}$. The algorithm will output an n-dimensional real-valued vector $f = [f_l^T f_u^T]^T = (f_1, \dots, f_l, f_{l+1}, \dots, f_{n=l+u})^T$, which can generate a threshold value to perform the label predictions on $(f_1, \dots, f_n)$ as a result of the learning. The label information propagates from the (labeled) node to an (unlabeled) node when they are coupled by a path of high density (e.g., the value of w_{ij} is large), their outputs $f's$ are likely to be close, whereas their outputs need not be close if they are separated by a low-density region (e.g., the value of w_{ij} is small). These assumptions are reflected in the value of f by minimizing the following quadratic function [13, 21]:

$$\min_f (f - y)^T (f - y) + \mu f^T L f$$

where L is the graph Laplacian, defined as L = D - W where $D = diag\,(d_i), d_i = \sum_i w_{ij}$. The parameter μ trades off loss and smoothness. Thus, the solution of this problem becomes

$$f = (I + \mu L)^{-1} y.$$

Based on the basic framework of graph-based SSL, SSL Co-training obtains more labeled data by assigning labels to unlabeled data, i.e., "pseudo-labels," and uses them for model learning as if they were labeled [8]. The model involves multiple member models where pseudo-labels are determined based on agreements among the members. Therefore, it is named as SSL Co-training. The toy example shown in Figure 1 is helpful for understanding the model. At the start of the algorithm, each of the member models (for simplicity, we assume two classifiers) is trained on the original graph (a). After training, both member models produce predicted labels for the unlabeled nodes. The unlabeled nodes are pseudo-labeled when the member models agree on labeling, or it remains unlabeled. The resulting graph is shown in (c). SSL Co-training increases the performance of an ordinary SSL thanks to the pseudo-labeled data points. Further details on the method can be found in [8].



(a) Original SSL graph (b) SSL Co-training: Predicted labels (c) SSL Co-training: Pseudo labeled graph

Fig. 1. Original SSL Graph and SSL Co-Training: (a) In graph-based SSL, the labeled nodes are represented by '+1(survived)' and '-1(dead)', whereas unlabeled nodes are represented by '?'. The two figures, (b) and (c), provides schematic description of SSL Co-training.

2.2 Variable Importance Calculation with Decision Trees

After prediction, we obtain information on the importance of the variables used as input for the survivability prediction. The patient samples with predicted labels by SSL Co-training are fed to decision trees (DT) and re-classified. DT can provide interpretability on what happened during prediction, i.e., 'which variables are most significantly contributed to survival/death classification?'. An answer for the question is naturally obtained by reclassifying the (variables, the predicted label) pairs of patient samples. Note that neither a validation nor a test set is used since the DT here is only employed for the purpose of description on the output of the prediction, not for prediction itself.

During the training, DT recursively splits samples in a root node into two or more subgroups until a final tree is constructed. While a tree is growing, it identifies a splitting variable and corresponding threshold value that maximizes the homogeneity of the resulting two or more subgroups of samples. The issue of variable importance is related to the splitting criteria of DT. The most well-known criteria includes Gini index (used in CART) [17], Entropy based information gain (used in ID3, C4.5, C5) [16], and Chi-squared test (used in CHAID) [18]. There are some differences among those criteria, the commonly used measure of importance is based on the surrogate splits $\tilde{s}_x$. It computes the improvement in homogeneity by the splitting of variable x, $\Delta I(\tilde{s}_x, t)$, at each node t in the final tree, $t \in T$. Then, the measure of importance $\mathrm{M}(x)$ of variable x is defined as the sum across all splits in the tree of the improvements that x has when it is used as a primary or surrogate splitter [11, 17, 22]:

$$\mathrm{M}(x) = \sum_{t \in T} \Delta I(\tilde{s}_x, t).$$

Since only the relative magnitudes of the $\mathrm{M}(x)$ are interesting, the actual values of variable importance are the normalized quantities. The most important variable then has value 1, and the others are in the range 0 to 1.

$$\mathrm{VI}(x) = \frac{\mathrm{M}(x)}{\max_x \mathrm{M}(x)}. \quad (1)$$

3 Experiments

3.1 Experimental Setting

The SEER database claims to have one of the most comprehensive collections of cancer statistics [1, 15]. The data consists of 162,500 records with 16 input variables and one target class variable. The input variables include incidence, mortality, prevalence, survival, lifetime risk, and statistics by race/ethnicity, etc. The target variable 'survivability' is a binary categorical feature with values '–1' (not survived or dead) or +1 (survived). The first three columns of Table 1 summarizes the variables and the corresponding descriptions.

Table 1. Prognostic elements of breast cancer survivability (SEER) and the resulting ranking by the order of variable importance

Prognostic elements		Description	Variable Importance
1	**Lymph Node Involvement**	None, (1–3) Minimal, (4–9) Significant, etc.	**1.00**
2	**Stage**	Defined by size of cancer tumor and its spread	**0.87**
3	**Site Specific Surgery**	Information on surgery during first course of therapy, whether cancer-directed or not.	**0.77**
4	**Number of Positive Nodes Examined**	When lymph nodes are involved in cancer, they are known as positive.	**0.75**
5	**Tumor Size**	2–5 cm; at 5 cm, the prognosis worsens	**0.61**
6	**Age at Diagnosis**	Actual age of patient in years	**0.44**
7	**Clinical Extension of tumor**	Defines the spread of the tumor relative to the breast	**0.38**
8	**Number of Nodes Examined**	The total number of (positive/negative) lymph nodes that were removed and examined by the pathologist.	**0.12**
9	**Histological Type**	Form and structure of tumor	**0.12**
10	**Primary Site**	Presence of tumor at particular location in body. Topographical classification of cancer.	**0.07**
11	**Grade**	Appearance of tumor and its similarity to more or less aggressive tumors	**0.05**
12	**Marital Status**	Married, Single, Divorced, Widowed, Separated	**0.05**
13	**Race**	Ethnicity: White, Black, Chinese, etc.	**0.02**
14	**Number of Primaries**	Number of primary tumors (1–6)	**0.01**
15	**Behavior Code**	Normal or aggressive tumor behavior is defined using codes.	**0.00**
16	**Radiation**	None, Beam Radiation, Radioisotopes, Refused, Recommended, etc.	**0.00**
Survivability		Target binary variable defines class of survival of patient.	

The generalization abilities of five representative predictive models, i.e., DT, ANN, SVM, SSL, and SSL-Co training, were compared. For each of the five models, the best performance was selected by searching over the respective model-parameter spaces. The area under the receiver operating characteristic (ROC) curve (AUC) was used as performance measures. To avoid the difficulties in learning of the predictive

models, caused by the large-sized and class-imbalanced dataset, 40,000 data points were used for the training set and 10,000 for the test set, which were drawn randomly without replacement. The equipoise dataset of 50,000 data points was eventually divided into ten groups and five-fold cross validation was applied to each.

3.2 Results

Figure 2 shows a comparison of the AUC results of DT, ANN, SVM, SSL, and SSL Co-training in due order, for each of the 10 data sets. SSL Co-training produced an average AUC of 0.81, which was the best of the five models although comparable performance was delivered by SVM. On the other hand, DT showed an average AUC of 0.73, and just ranked the worst performed model ANN. Either DT or ANN may be a good predictive model for some other problems, but are less likely to be the one than other three models in the current study.

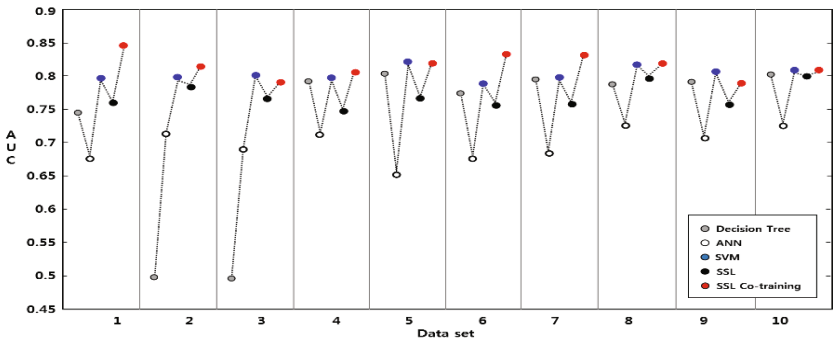


Fig. 2. Performance comparison over 10 data sets: DT, ANN, SVM, SSL, and SSL Co-training. The average AUCs are 0.73, 0.70, 0.80, 0.78, and 0.81, respectively.

The labels for the test samples were obtained from SSL Co-training, which performed best among the five competing models. The test samples with predicted labels were input to DT, together with the training samples.

The last column of Table 1 shows ranking of the 16 input variables in terms of Eq.(1), the relative magnitudes to the value of the most important variable. It shows that 'Lymph Node Involvement' is the most determinant variable in identifying survived/dead patients, therefore it has a value of 1. And in order of variable importance, 'Stage', 'Site-specific Surgery', 'Number of Positive Nodes Examined' and 'Tumor Size' belong to the top-tier ranked up to 5th variables, and are regarded as more important ones than the rest. The five variables are all related to the findings from a pathologic exam. It is known as the best way to assess lymph node status and can give a first estimate of breast cancer stage and the size of tumor.

To discern that how those variables influence to classification, the two patient groups of the survived and the dead were profiled, respectively. Figure 3 shows a graph of the average values of 16 variables for the two patient groups. The grey line

in the center stands for the overall average of the variable, whereas the blue/red line stands for the group average of the survived/dead, a scaled value relative to the overall average. Compared the averages of the two groups, significant differences can be found by 'Lymph Node Involvement', 'Number of Positive Nodes Examined', 'Stage', 'Behavior Code', 'Site-Specific Surgery', 'Tumor Size', 'Age at Diagnosis', whereas 'Marital Status' and 'Race' do not provide significant information on discriminating the two groups. Relatively, a general pattern of the survived patients is less involvement of lymph nodes, an earlier stage, a smaller sized tumor, non-invasive in cancer behavior, less (site-specific) surgeries, younger in terms of age at diagnosis. On the other hand, the dead patients show a pattern of larger spread of cancer over lymph nodes, a larger tumor size, more aggressive and invasive cancer behavior, more surgeries and radiation therapies, and an older age at diagnosis. This comparative profiling can help predict and understand the chances for long-term survival of the patients and also guide proper treatments that fit for each of the patients.

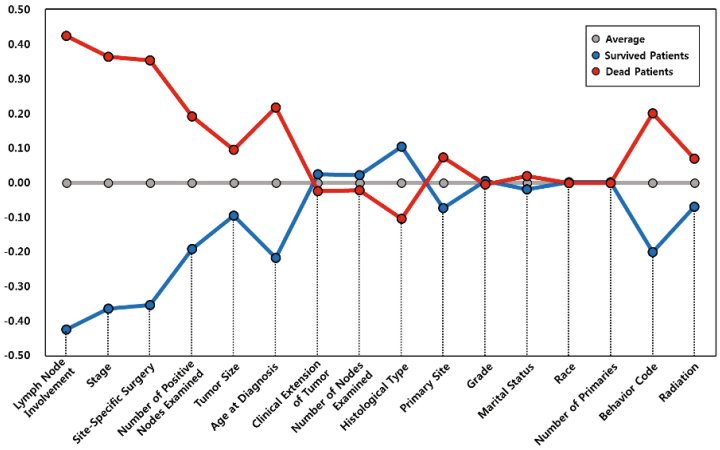


Fig. 3. Variable profiling for (a) survived patients and (b) dead patients

4 Conclusion

Although most of the up-to-date machine learning algorithms offer daily updated best predictions for survivability of the breast cancer patients, they seldom provide explicit explicability of which variable is the most significant during prediction. To unveil the implicit mechanism of the prediction procedure, an idea of embedding the procedure for variable importance calculation was proposed. The proposed model hybridizes two sub-models: (1) SSL Co-training classifies patients into two classes of the survived and the dead, and then (2) DT calculates the importance of prognosis factors. Knowing the significant variables will lead to better insights in cancer prognosis, and less time and cost by excluding redundant ones during data collection. The present study triggers possible future works. First, the proposed method is flexible in that any

winner model can be employed instead of SSL Co-training. Therefore, a further extension will be made on comparative study with other models. The hybrid approach for breast cancer prognosis is yet general and its full application for different cancer types will still require a continued refinement.

Acknowledgments. HS would like to gratefully acknowledge support from the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2013R1A1A3010440/2010-0028631).

References

1. American Cancer Society, http://www.cancer.org
2. National Cancer Institute. Breast Cancer Statistics, USA (2010), http://www.cancer.gov/cancertopics/types/breast (accessed July 11, 2011)
3. Sun, Y., Goodison, S., Li, J., Liu, L., Farmerie, W.: Improved breast cancer prognosis through the combination of clinical and genetic markers. Bioinformatics 23, 30–37 (2007)
4. Khan, U., Shin, H., Choi, J.P., Kim, M.: Wfdt - Weighted Fuzzy Decision Trees for Prognosis of Breast Cancer Survivability. In: Roddick, J.F., Li, J., Christen, P., Kennedy, P.J. (eds.) The Proceedings of the Seventh Australasian Data Mining Conference, Glenelg, South Australia, pp. 141–152 (2008)
5. Delen, D., Walker, G., Kadam, A.: Predicting breast cancer survivability: a comparison of three data mining methods. Artificial Intelligence in Medicine 34, 113–127 (2005)
6. Cruz, J.A., Wishart, D.: Applications of Machine Learning in Cancer Prediction and Prognosis. Cancer Informatics 2, 59–78 (2006)
7. Maglogiannis, I., Zafiropoulos, E., Anagnostopoulos, I.: An intelligent system for automated breast cancer diagnosis and prognosis using SVM based classifiers. Applied Intelligence 30, 24–36 (2009)
8. Kim, J.H., Shin, H.: Breast Cancer Survivability Prediction using Labeled, Unlabeled, and Pseudo-Labeled Patient Data. Journal of the American Medical Informatics Association 20, 613–618 (2013)
9. Shin, H., Kim, D., Park, K., Ali, A.: Breast Cancer Survivability Prediction with Surveillance, Epidemiology, and End Results Database. In: The Proc. of Translational Biomedical Conference, Seoul, Korea (2011)
10. Dursun, D., Glenn, W., Amit, K.: Predicting breast cancer survivability: a comparison of three data mining methods. Artificial Intelligence in Medicine 34, 113–127 (2005)
11. Matignon, R.: Data Mining Using SAS Enterprise Miner. John Wiley & Sons, Inc. (2007)
12. Olaru, C., Wehenkel, L.: A complete fuzzy decision tree technique. Fuzzy Sets and Systems 138, 221–254 (2003)
13. Shin, H., Hill, N.J., Lisewski, A.M., Park, J.S.: Graph sharpening. Expert Systems with Applications 37, 7870–7879 (2010)
14. Shin, H., Lisewski, A.M., Lichtarge, O.: Graph sharpening plus graph integration: a synergy that improves protein functional classification. Bioinformatics 23, 3217–3224 (2007)
15. SEER, Surveillance, Epidemiology and End Results program National Cancer Institute (2010), http://www.seer.cancer.gov (accessed July 11, 2011)
16. Quinlan, J.: C4.5: Programs for machine learning. Morgan Kaufmann, San Mateo (1993)

17. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall/CRC, New York (1984)
18. Kass, G.V., Gordon, V.: An Exploratory Technique for Investigating Large Quantities of Categorical Data. Applied Statistics 29, 119–127 (1980)
19. He, J., Carbonell, J., Liu, Y.: Graph-Based Semi-Supervised Learning as a Generative Model. In: Veloso, M.M. (ed.) The Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, pp. 2492–2497 (2007)
20. Chapelle, O., Schölkopf, B., Zien, A.: Semi-Supervised Learning. The MIT Press, Cambridge (2006)
21. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and Semi-supervised Learning on Large Graphs. In: Shawe-Taylor, J., Singer, Y. (eds.) COLT 2004. LNCS (LNAI), vol. 3120, pp. 624–638. Springer, Heidelberg (2004)
22. Guyon, I., Elisseeff, A.: Introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)

Protein Structure Prediction with a New Composite Measure of Diversity and Memory-Based Diversification Strategy

Rumana Nazmul and Madhu Chetty

Gippsland School of Information Technology, Monash University, Australia
{rumana.nazmul,madhu.chetty}@monash.edu

Abstract. Protein structure prediction (PSP) problem is a multimodal problem that can be tackled efficiently by evolutionary algorithms. However, evolutionary algorithms often fail to find the global optima due to genetic drift while solving the complex problems with a lot of peaks in the fitness landscape. Therefore, the need to efficiently measure as well as maintaining population diversity has significant effects in performance of evolutionary algorithms. In this paper, we introduce a composite measure of population diversity by hybridizing the phenotypic properties along with the distribution of individuals in a population over the fitness landscape. We further propose a memory-based diversification technique for the maintenance and promotion of diversity to prevent occurrence of stuck condition in multimodal problems such as PSP. Experiments conducted on protein structure prediction with HP benchmark sequences for 3D cubic lattice model illustrate that the proposed techniques are useful in improving the optimization process in terms of convergence as well as for achieving the optimal energy.

Keywords: Protein Structure Prediction, Diversity, Phenotype.

1 Introduction

The success of searching for the global optima through the complex fitness landscape of a multimodal problem having several peaks, greatly depends on both the search approach as well as on the search space. While dealing with the computational complexity of a long-standing NP-hard problem [1] like PSP, even with simplified lattice models [2], deterministic algorithms cannot provide satisfactory solutions in reasonable amount of time [3]. Consequently, it has motivated the investigation of various stochastic search methods (e.g., evolutionary algorithms) such as genetic algorithms (GA) [4], memetic algorithms (MA) [3], estimation of distribution algorithms (EDA) [5] to address the problem. Among these evolutionary methods, MA that offers the goodness of both local and global search algorithms, can be more robust than other approaches to deal with the complex and challenging nature of PSP. However, like other evolutionary approaches, the problem of getting stuck in local optima at an early stage of evolution due to

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 649–656, 2013.

the lack of diversity is frequently observed in MA. Hence, in multimodal problems, it is necessary to maintain a sufficiently diverse population of potential solutions so that the search can trail different paths through the landscape to explore it efficiently. It is therefore considered as a key factor in success of MA to measure and subsequently preserve required level of diversity in the population. Diversity usually refers differences in structures [6] or behaviors [7, 8] of individuals in a population. In this paper, to address the issue associated with the aim of maintaining diversity, we propose a new composite measure of diversity based on phenotypic and distribution of individuals over the fitness landscape. Further, we propose a diversification mechanism to maintain the required level of diversity in the population when it reaches below a certain threshold value. The efficacy of the proposed techniques has been investigated with the protein structure prediction problem for benchmark sequences of 3D cubic HP model.

2 Background

2.1 HP Lattice Model

Protein structure prediction (PSP) problem can be defined as the problem of finding the native structure of a protein having the lowest possible free energy given only its amino acid sequence. The Hydrophobic-Polar (HP) model [2] is the most widely used model for lattice simulation that considers the hydrophobic effect as the main driving forces for the formation of protein structure. According to the model, amino acids are classified either as *hydrophobic (H) or polar (P)* based on their affinity for water. More details about the model and the energy function used in this paper can be found in [3, 9].

2.2 Existing Measures of Diversity

In the field of evolutionary computation, the pre-requisite of maintaining diversity is to measure it by a metric based on some population features such as individual fitness values (phenotype), structures (genotype) or the combination of two [6]. Among these, the most common approach is to quantify the structural varieties in the population by hamming distance which calculates the distance between two individuals by the number of bitwise differences (moves, in our case) they have. However, phenotypic or behavioral measures have the following advantages over the genotypic one: i) The phenotypic diversity usually implies genotypic diversity, but the reverse is not necessarily true ii) Phenotypic measure does not depend on a particular encoding scheme iii) Due to the straightforwardness of phenotypic diversity measure, it requires less time in computation [6]. Measuring diversity based on phenotype has been proposed in [7, 8, 10, 11].

Furthermore, the extreme disorder or uneven distribution of population can be considered as an indication of premature convergence [12]. However, the limitation of all the above measures is that these do not convey enough information about the distribution of fitness values over the fitness landscape and none of these measures can differentiate among the different diverse situations occurring through the evolutionary process.

2.3 Existing Diversification Techniques

When the optimization process gets stuck in local minima, the number of the sub-optimal solutions increases over the generations due to the selection pressure and spreads over the entire population resulting in loss of diversity [6]. Consequently, it leads to premature convergence affecting further improvement. Hence, it is necessary to trigger a mechanism to increase the level of diversity when it is dropped below a certain threshold value or the evolution is stagnant for specified number of generations. The most conventional diversification strategy is to replace certain percentage of individuals from the population with new randomly generated individuals either i) in every generation [13] or ii) whenever current diversity goes below a pre-defined threshold value [14]. In most cases, randomly generated individuals replace the existing ones that are selected either randomly or based on specific criteria such as genotypic similarity or the fitness values.

3 Proposed Method

3.1 Novel Composite Metric for Measuring Diversity

To appraise the diversity of population we have proposed a novel metric to measure the diversity based on the phenotypic feature of population. The proposed measure not only highlights on the phenotypic property, also capable of differentiating among various situations with different distribution of fitness values over the fitness landscape. Therefore, our proposed composite measure of diversity comprises of two terms: for any population, the first term (T_1) measures the phenotypic differences among individuals and the second term (T_2) evaluates the distribution of individuals over the fitness landscape.

A. Phenotypic Measure: To measure the diversity in any G_t^{th} generation, we partition the population into equidistant groups called *fitness bucket* according to their fitness values. If f_t^{worst} and f_t^{best} respectively denote the worst and the best fitness values in the t^{th} (current) population, and ϵ represents the difference between the lower and upper limit value of each bucket, the number of possible *fitness buckets* (denoted by F_t) is obtained as $(\lceil f_t^{best} \rceil - \lfloor f_t^{worst} \rfloor)/\epsilon$. Here, the i^{th} bucket contains the individuals with fitness values in the range $(f_{t,i}^{min}, f_{t,i}^{max}]$, where $f_{t,i}^{max} - f_{t,i}^{min} = \epsilon$. The number of buckets containing at least one individual is denoted as $F_{available}$. Finally, F_{max} represents the number of maximum possible *fitness bucket* up to the G_t^{th} generation, obtained by $(\lceil f_t^{best} \rceil - \lfloor f_0^{worst} \rfloor)/\epsilon$. The first term or phenotypic measure is computed according to the following equation:

$$T_1 = \frac{(F_t - F_{available}) \times \frac{F_{available}}{F_t} + F_{available}}{F_{max}} \tag{1}$$

Although the diversity can be measured by $F_{available}/F_{max}$ representing the ratio of available fitness buckets and maximum possible fitness buckets, the other term in numerator is used to represent the distribution of these fitness buckets over the fitness landscape. For example, if we compare between two populations

both having the same number of possible fitness buckets (F_t) but with different number of occupied fitness buckets ($F_{available}$), the population having more occupied buckets must be considered more diverse than the other one. On the contrary, while comparing two populations having same number of occupied fitness buckets in different span (i.e., the number of possible fitness buckets F_t), the population with larger span is considered as more diverse since the occupied fitness buckets are more sparse here. The term $(F_t - F_{available}) \times \frac{F_{available}}{F_t}$ is used to balance between the two different aforementioned situations. In brief, the proposed term quantifying phenotypic diversity not only measures diversity based on fitness values, but also considers the distribution of these values. To comprehend the scenario clearly, let us consider Table 1, that shows three different scenarios for varying F_{max}, F_t and $F_{available}$ values:

Table 1. Different scenarios of T_1, in terms of numerical values, for two different F_{max} values with corresponding various $F_{available}$ and F_t

F_{max}	15						20					
F_t	9	9	10	10	15	15	9	9	10	10	20	20
$F_{available}$	2	5	9	10	10	15	2	5	9	10	10	20
T_1	0.24	0.48	0.66	0.67	0.89	1.00	0.18	0.36	0.50	0.50	0.75	1.00

B. Fitness Based Distribution of Population (DoP): To measure the DoP, as a first step, we sum up the absolute values of deviation of number of individuals actually contained in each *fitness bucket* (n_i) from the number ($n_b = N/F_{available}$, where N denotes the population size) residing in the *fitness bucket* in an evenly distributed population.

$$DoP_{sum} = \sum_{i=1}^{F_{available}} |n_b - n_i| \tag{2}$$

Further, the value of DoP_{sum} is normalized to [0, 1] by dividing it with a value corresponding to the worst case scenario. The worst case calculation is performed according to the following equation by considering $F_{available}$-1 buckets each with only one individual and one bucket containing the remaining individuals:

$$DoP_{worst} = |n_b - 1| \times (F_{available} - 1) + |n_b - (N - (F_{available} - 1))| \tag{3}$$

To get the final value of DoP_{norm}, we divide Eq. 2 by Eq. 3

$$DoP_{norm} = \frac{\sum_{i=1}^{F_{available}} |n_b - n_i|}{|n_b - 1| \times (F_{available} - 1) + |n_b - (N - (F_{available} - 1))|} \tag{4}$$

As Eq. 4 is a minimization function (the more individuals are distributed evenly, the lower the value of DoP_{norm} is), we subtract DoP_{norm} from 1 to obtain the final value for T_2 in terms of maximization function:

$$T_2 = 1 - DoP_{norm} \tag{5}$$

C. Composite Metric to Measure Diversity: Finally, diversity of the population (Dv) is calculated as follows:

$$Dv = w_1 \times T_1 + w_2 \times T_2 \tag{6}$$

where, w_1 and w_2 are two weighing factors. To execute predominance of T_1 over T_2 , the value of w_1 should be greater than w_2 and has been set based on empirical observations. It should be noted that, the complexity of Hamming-distance based diversity measurement technique is $O(N^2L)$, which is NL (L=length of the sequence) times higher than the proposed technique.

D. Novel Memory-Based Diversification Strategy: Although, the traditional random immigrant scheme maintains the diversity level of the population, it may have no actual effect in later phases of evolution. This is due to the fact that, for survival, these individuals may have to compete with the existing individuals that are more fitter than the random immigrants. Here, we propose a new diversification strategy using memory-based immigrants with a memory size $m = x\% * N * div\_interval$, where x is the percentage of immigrants during the diversification, N is the size of the population and $div\_interval$ denotes the frequency at which diversification procedure is invoked. This memory is used to store unexploited individuals having different fitness values generated by the cross-over operation at every generation during the course of evolution. These individuals are used as immigrants to diversify the population in later stages. With respect to which individuals should be stored in the memory, we consider the offspring with worse fitness value than the other one generated by a cross-over operation, since it has not yet been exploited in the optimization. At every generation, exactly one individual having an unique fitness value is selected to be stored into the memory. Once the memory is full, we insert an individual with fitness value f_{new} if any other individual with the same fitness values does not exist in the memory. To accommodate the individuals with f_{new} in the memory, we replace the most recently inserted individuals from the crowded region, i.e., the maximum number of individuals with the same fitness value. The rationale behind this is to use the individuals from memory as immigrants with sufficient generation difference to prevent the co-existence of offspring generated from the same parents. The proposed technique ensures the inclusion of completely new and unexploited individuals as immigrants for diversifying the population. Further issues as to when and how to use the memory, is described below.

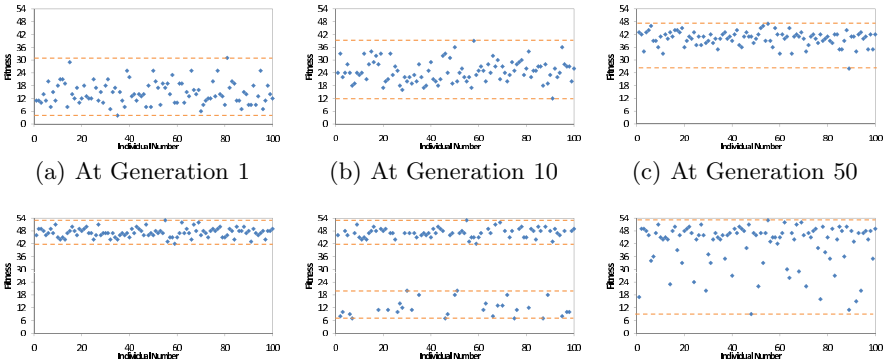
In order to apply the diversification, we check the diversity of the population in every DV_F generation, and if the diversity of a population (Dv) is less than the threshold ψ_t and the best fitness of the population is not changing for the last h generations, we apply the proposed diversification technique as follows:

- $x\%$ individuals of the population are selected probabilistically (using fitness-based probability) from the memory having fitness values in the range $(\lfloor f_0^{Worst} \rfloor - \lceil f_t^{Best} \rceil)$ giving preferences to relatively fitter individuals.
- In order to accommodate the incoming immigrants in the population, we select the individuals from existing fitness buckets for replacement. While

selecting the individuals to be replaced, the crowded buckets (according to the DoP) are given preference.
– While selecting an immigrant from the stored individuals with the same fitness value in the memory, we apply First-in-First-out (FIFO) technique to prevent the coexistence of closely similar individuals.

4 Experimental Results and Discussion

To evaluate the performance of the proposed and random diversification technique, we consider all non-benchmark and benchmark sequences defined in [3], which are widely used to testify the methods for realizing 2D and 3D HP lattice models. The evaluation of the proposed diversification has been demonstrated for the complicated sequences (B5-B9) having moderate length (i.e., 48 to 85). To get an empirical observation on the influence of the proposed and random diversification, we have executed a sample run of 100 generations without any diversification. In 100^{th} generation, we have separately applied the proposed and random diversification over the population. Fig. 1 (a)-(d) shows the snapshot of individuals in the population over different generations (i.e., 1, 10, 50, 100). At 100^{th} iteration, we have separately applied both the random and proposed diversification, where the population after diversifications are shown in Fig. 1(e) and Fig. 1(f), respectively. First four snapshots in Fig. 1 clearly shows the gradual decrease of diversity over the generation and the span of energy values has become narrow in 100^{th} generation. Although, the diversity of the population has been increased after applying the random diversification, it divides the population into two visible spans. On the other hand, the distribution of population is found to be excellent after applying the proposed diversification, as shown in Fig. 1(f). Further, we have investigated the effect of proposed diversification in protein structure prediction with a simple Memetic Algorithm (MA) where both proposed and existing diversification techniques have been used separately for 3D HP lattice. Other than the diversification algorithms, the cross-over and mutation rate used here are 0.8 and 0.1, respectively. Here we have applied pull-move as local search [3] with the rate 0.3 over the individuals of the population in every 30 generations interval. In both the diversification techniques, 30% of total population are selected for replacement. The selected individuals are then replaced by randomly generated individuals in the existing diversification technique, whereas in the proposed technique, previously generated but unexploited solutions, which are retrieved from memory are used as immigrants. Both the MAs are executed for 25 runs with maximum 1,00,000 fitness evaluations in each runs. For five benchmark functions ($B5-B9$), we plot the obtained energy values for each modules in Table 2. We observe that, for all the sequences, the MA with proposed diversification technique ends up the final $1{,}00{,}000^{th}$ fitness evaluation with better fitness values than the MA with random diversification. The average energies in the MA with proposed diversification technique are found superior than the MA with random diversification.



(a) At Generation 1 (b) At Generation 10 (c) At Generation 50

(d) At Generation 100 (e) After Random Diversification (f) After Proposed Diversification

Fig. 1. Effect of Random and proposed Diversification. Snapshots of individuals in the population at generation (a) 1 (b) 10 (c) 50 and (d) 100 (before diversification). Individuals in the population after (e) Random and (f) Proposed diversification. For better viewing perspective, absolute values of energies are considered in the Y-axis

Table 2. Best energies reported by MAs with proposed and random diversification. E^* for each benchmark sequence numbers indicate the known best energies.

	With Proposed (Memory based) Diversification		**With Existing (Random) Diversification**	
BN(E^*)	Best	Avg±STD	Best	Avg±STD
B5(-31)	-29	-28.2 ±1.10	-25	-24.0 ±0.71
B6(-32)	-28	-26.8 ±0.84	-24	-22.6 ±1.52
B7(-54)	-47	-45.6 ±0.89	-45	-43.6 ±1.14
B8(-58)	-49	-46.2 ±2.17	-42	-40.4 ±0.89
B9(-79)	-68	-65.6 ±1.52	-63	-61.4 ±1.52

5 Conclusion

In this paper, we have proposed a new metric to measure the diversity of a population based on its phenotypic feature along with the distribution of the individuals over the fitness landscape. Moreover, we have also introduced a novel memory-based diversification mechanism that ensures required level of diversity in the population. An extensive analysis is performed to assess the efficacy of the proposed measure of diversity and diversification technique with widely used benchmark protein sequences, that has shown excellent performance.

References

1. Crescenzi, P., Goldman, D., Papadimitriou, C.H., Piccolboni, A., Yannakakis, M.: On the complexity of protein folding. In: RECOMB, pp. 61–62 (1998)
2. Lau, K., Dill, K.A.: A lattice statistical mechanics model of the conformational and sequence spaces of proteins. Macromolecules 22(10), 3986–3997 (1989)

3. Islam, M.K., Chetty, M.: Clustered memetic algorithm with local heuristics for ab initio protein structure prediction. IEEE Trans. Evolutionary Computation 17(4), 558–576 (2013)
4. Unger, R., Moult, J.: Genetic algorithm for protein folding simulations. Journal of Molecular Biology 231(1), 75–81 (1993)
5. Santana, R., Larranaga, P., Lozano, J.A.: Protein folding in simplified models with estimation of distribution algorithms. IEEE Transactionson on Evolutionary Computation 12(4), 418–438 (2008)
6. Burke, E., Gustafson, S., Kendall, G.: Diversity in genetic programming: An analysis of measures and correlation with fitness. IEEE Trans. on Evolutionary Computation 8(1), 47–62 (2004)
7. Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for off-line and on-line control design of pmsm drives. IEEE Trans. on Systems Man and Cybernetics Part B, Special Issue on Memetic Algorithms 37(1), 28–41 (2007)
8. Tirronen, V., Neri, F., Karkkainen, T., Majava, K., Rossi, T.: A memetic differential evolution in filter design for defect detection in paper production. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 320–329. Springer, Heidelberg (2007)
9. Nazmul, R., Chetty, M., Samudrala, R., Chalmers, D.: Protein structure prediction based on optimal hydrophobic core formation. In: IEEE Congress on Evolutionary Computation, pp. 1–9 (2012)
10. Neri, F., Toivanen, J., Cascella, G.L., Ong, Y.S.: An adaptive multimeme algorithm for designing hiv multidrag therapies. IEEE-ACM Trans. on Comput. Biology and Bioinformatics 4(2), 264–278 (2007)
11. Neri, F., Toivanen, J., Makinen, R.: An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for hiv. Applied Intellegence, Special Issue on Cdicine and Biology (2007)
12. Rosca, J.: Entropy driven adaptive representation. In: Proceedings of the Workshop on Genetic Programming: From Theory to Real world Application, pp. 23–32 (1995)
13. Grefenstette, J.J.: Genetic algorithms for changing environments. In: Parallel Problem Solving From Nature II, pp. 137–144 (1992)
14. Wang, H., Wang, D., Shengxiang, Y.: A memetic algorithm with adaptive hill climbing startegy for dynamic optimization problems. Soft Comput. 13(8-9), 763–780 (2009)

An Algorithm for Parallelizing Sequential Minimal Optimization

Xinyue Wang and Jun Guo

Computer Center
East China Normal University
3663 Zhong Shan Rd. N., Shanghai, China
jguo@cc.ecnu.edu.cn

Abstract. In this paper, a new algorithm for training support vector machines (SVMs) for classification problems with parallel sequential minimal optimization (SMO) is proposed. The selection of the working set is paralleled so that the iteration of the optimization process is reduced greatly. The experimental results show the training time of the proposed method is always less than the original SMO algorithm, and at the same time the classification accuracy is kept.

Keywords: SVM, SMO, Parallelization.

1 Introduction

Support Vector Machines (SVM) is a very useful tool for solving pattern recognition problems [1]. Due to its good performance in generalization on the basis of statistical learning theory [2], more and more research on SVM has been done and a lot of variants of SVM have been implemented.

The training step of SVM in the dual space was proposed in [1] using Lagrange multipliers. With the increasing number of the training data and support vectors, more optimization strategies have been adopted, especially some decomposition methods such as chunking[1], Osuna's algorithm, and Sequential Minimal Optimization(SMO) [3, 4].

The chunking method starts with a subset of data called chunks and iteratively enlarges that subset by including those examples that violate the optimization conditions. In Osuna's algorithm, the training examples were separated into two groups: the working set denoted as B and the rest of the data defined as N. In [4], Osuna's theorem demonstrates that moving a variable from set B to set N does not change the cost function and ensures that there is a strict improvement in the cost function after moving a variable that violates the optimality condition. Osuna's algorithm makes training SVMs tractable especially when the number of support vectors is quite large. SMO [3] ensures its convergence due to Osuna's theorem and optimizes only two Lagrange multipliers at a time, which means that the size of working set is only two. The performance of SMO in training time is better than Osuna's algorithm according to most

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 657–664, 2013.

of the experiments in [3]. This leads to a better performance in many circumstances and can be applied to both classification and regression problems. There are also some other algorithms using decomposition methods like SVM$^{\text{Light}}$[5] and the algorithm used in LIBSVM [6, 7]. The former utilizes a first-order approximation of the object function and seeks a steepest direction of descent. The latter considers a more precise approximation using second order information, aiming at obtaining a faster convergence speed [6].

There are some researches about the parallelizing SMO algorithm, reducing the training time of SVM considerably. In [8], the whole data set is divided into smaller subsets and distributed to different processors. By parallelizing the procedure of updating the error array denoted by F_i, the parallel SMO algorithm is much faster than the original one, especially when the number of data is large. The Casade SVM was introduced in [9], which trains SVMs in layers like filters. Because some examples are unlikely to be support vectors in an early stage of optimization, they are filtered. Those being support vectors are passed down to another layer for further training and testing. The process iterates until convergence. The main idea in parallel SVM is to reduce the time and memory cost by training, which derives our parallel algorithm dealing with working set selection.

In this paper, an algorithm of parallelizing the procedure in training SVM is proposed. We focus on the selection of working sets and choose two pairs at the same time. From the experiment, we show that the time of training is less than the original serial SMO. Furthermore, the prediction accuracy is maintained.

This paper is organized as follows: section 2 gives a brief introduction to SVM, decomposition methods and some related works. In section 3, we propose our parallelizing solutions and the experiment results are shown in section 4. Finally, section 5 discusses some problems in our method and gives a possible solution in future works.

2 Support Vector Machines

In classification problems, we denote $\{X_i, y_i\}_{i=1}^{l}$ as input samples where $\{X_i\}_{i=1}^{l}$ are the training examples and $y_i = \{-1,1\}_{i=1}^{l}$ are the corresponding labels. Training an SVM aims at solving an optimization problem given below:

$$\min_{w} \quad f(w) = \frac{1}{2} w^T w \tag{1}$$

$$\text{subject to } y_i(w^T x_i + b) \geq 1 \quad i = 1,2,...,l$$

In order to handle misclassifications when the input samples contain noises, slack variables ξ_i are added into (1). Thus we obtain a soft margin classification problem.

$$\min_{w} \quad f(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \tag{2}$$

$$\text{subject to} \quad y_i (w^T x_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0, \ i = 1, 2, ..., l$$

where C > 0 is the regularization parameter. One way of solving (2) is to handle with the following dual problem based on Lagrange duality theorem and get the following optimization problem in the dual.

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \varphi^T(x_i) \varphi(x_j) \tag{3}$$

$$\text{subject to} \quad \sum_{i=1}^{l} \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C \ \ i = 1, 2, ..., l$$

where α_i is the Lagrange multiplier and $\varphi(x)$ is the mapping function and we denote $K(x, x') = \varphi^T(x) \varphi(x')$ known as the kernel function which implicitly maps data to a high-dimensional space without computing the mapping function. There are some typical kernel functions such as polynomial kernel and Gaussian kernel. Finally, if the kernel matrix satisfies Mercer's condition which means that K is positive semi-definite, (1) becomes a QP optimization problem which contains no local minima.

3 Sequential Minimal Optimization

However, the Kernel matrix $y_i y_j K(x_i, x_j)$ may be too costly to be fit into the memory due to its size and density. Platt's sequential minimal optimization (SMO) [3] focuses on choosing two instances as working set. However, due to its underlying inefficiency in choosing the threshold value, modifications of Platt's SMO are proposed in [10], which introduces an improvement by choosing and updating the worst violating pair and also deals with two threshold parameters. The optimization problem is defined as follows. First, we rewrite the optimization problem from (3).

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{4}$$

$$\text{subject to } \sum_{i=1}^{l} \alpha_i y_i = 0$$

$$0 \le \alpha_i \le C \quad i = 1, 2, ..., l$$

According to [10], we define:

$$I_{up}(\alpha) = I_0(\alpha) \bigcup I_1(\alpha) \bigcup I_2(\alpha)$$

$$I_{low}(\alpha) = I_0(\alpha) \bigcup I_3(\alpha) \bigcup I_4(\alpha)$$

where

$$I_0(\alpha) = \{i : 0 < \alpha < C\}$$
$$I_1(\alpha) = \{i : y_i = 1, \alpha_i = 0\}$$
$$I_2(\alpha) = \{i : y_i = -1, \alpha_i = C\}$$
$$I_3(\alpha) = \{i : y_i = 1, \alpha_i = C\}$$
$$I_4(\alpha) = \{i : y_i = -1, \alpha_i = 0\}$$

The optimization condition for the problem is the following:

$$\min_{i \in I_{up}(\alpha)} F_i(\alpha) \ge \max_{j \in I_{low}(\alpha)} F_j(\alpha) \tag{5}$$

where $F_i(\alpha) = y_i \frac{\partial Q(\alpha)}{\partial \alpha_i} = y_i (\sum_{j=1}^{l} \alpha_j y_i y_j K(x_i, x_j) - 1)$. Thus, if a pair of indices (i, j) violates the above optimization condition, it is called a violating pair. Due to [10], the optimality holds if and only if no violating pair exists.

4 Parallelizing the Process of Working Set Selection

In this section, the parallel algorithm is proposed. First, we briefly introduce the idea of our algorithm and its difference from some other solutions. Then we give a more detailed picture of how to perform our parallelization on choosing violating pairs in working set selection.

Different from the methods above, we parallelize the process of working set selection and the updates of Lagrange multipliers in LIBSVM. In one of the parallelizing processes, we choose the maximal violating pair and we also choose the second maximal violating pair in the other one. Thus we make the time spent on training decreased in classification problems. Different from [8], our method is not similar to a single program multiple data model (SPMD) because we choose different working sets in the two parallel processes. Also, our algorithm does not require interaction between the two parallelizing processes because we simultaneously choose the maximal and the second maximal violating pairs and thus do not use the information from each other until they complete their updates.

Then, we provide a description of the parallel algorithm. Recall that problem (3) is to be solved. Based on [6, 7], we are going to solve the object function using its second order information. After initialization, the parallelization phase begins. We denote the maximal violating pair as α_i, α_j and the second maximal violating pair as α_m, α_n. The training algorithm works as follows:

```
Algorithm 1:
begin
  Initialize step:
  Initialize gradient ∇Q(α) and Lagrange multipliers
  αi, i = 1….l,
  where l is the total number of training instances
  repeat
  Parallel step:
    1: Select working set and choose the maximal violating
    pair αi and αj
    2: Select working set and choose the second maximal
    violating pair αm and αn
  Serialization step:
    Update the status of αi and αj
    Update the status of gradient ∇Q(αi) and ∇Q(αj)
    Update the status of αm and αn
    Update the status of gradient ∇Q(αm) and ∇Q(αn)
    Compute the object value
  until the optimization criteria are met
Calculate the object value and output the decision func-
tion
end.
```

5 Experiments

The algorithm is implemented on LIBSVM and it shows that the algorithm approximately halves the time cost by the serial LIBSVM algorithm. Furthermore, the loss in accuracy of the parallel algorithm is acceptable. The data sets used are from the web page of LIBSVM [13]. The first data set is the Adult data set, which contains 14 features originally and is preprocessed according to [4]. The number of training examples range from 1605 to 16100 and the number of test examples range from 16461 to 30956. It shows that there is a speedup at almost 50%. We also examine data sets from [3], which show nearly the same result as the Adult data set. We also test our algorithm on small data sets such as splice from Delve and SVMguide from [11], obtaining results similar to the above ones.

Table 1. Experiments on data sets with test examples

Data Sets	Size of training examples	Size of test examples	Accuracy (LIBSVM / Parallel SVM)	Time Elapsed (LIBSVM / Parallel SVM) (ms)
a1a(UCI)	1605	30956	84.4166% / 84.2422%	483 /220
a2a(UCI)	2265	30296	84.592% / 84.5887%	936 /562
a4a(UCI)	4781	27780	84.5392% / 84.5887%	3588 / 1482
a5a(UCI)	6414	26147	84.4227% / 84.4418%	6599 / 3197
a6a(UCI)	11220	21341	84.4806% / 84.4853%	20561/ 11451
a7a(UCI)	16100	16461	84.8065% / 84.7336%	42292/20878
w1a (JP98a)	2477	47272	97.9121% / 97.9311%	297 / 109
w2a (JP98a)	3470	46279	98.0704% / 98.0704%	546 / 298
w3a (JP98a)	4912	44837	98.2314% / 98.1823%	904 / 544
w4a (JP98a)	7366	42383	98.1667% / 98.1667%	1591 / 998
w5a (JP98a)	9888	39861	98.2063% / 98.1987%	2543 / 1219
w6a (JP98a)	17188	32561	98.4184% / 98.4184%	8065 / 4322
splice(Delve)	1000	2175	88.5517% / 88.5517%	483 / 264
svmguide(CWH03a)	3089	4000	93.025% / 94.075%	4805 / 923

There are also some data sets which do not include specific data for testing. Thus, we split the whole data set into 7 pieces randomly and choose 6 of the 7 pieces of data for training and the rest one for testing. Because such preprocessing is done every time before running an experiment, the data is quite reliable. We average the results from the sum of the 10 experiments.

Table 2. Experiments on data sets without test examples

Data sets	Size of training examples	Accuracy (LIBSVM / Parallel SVM)	Time Elapsed (LIBSVM / Parallel SVM) (ms)
breast-cancer(scaled)	683	96.30712% / 96.18366%	9.4 / 9.3
diabetes(scaled)	768	76.38394% / 75.78623%	71.7 / 48.5
heart(scaled)	270	80.71581% / 81.27854%	26.6 / 9.5

From the table above, it shows that the time cost by LIBSVM is almost the same as the time cost by the parallel algorithm in breast-cancer data set from UCI. Actually, the above three data set does not contain a large number of examples, leading to short running time in both LIBSVM and parallel algorithm. The accuracy is also quite acceptable as we can see from above. In diabetes, the LIBSVM outperforms the parallel algorithm by less than 0.6% in prediction while they are very close to each other in the rest two data sets.

We next do our experiments on the MNIST data set which consists of 60000 instances for training examples and 10000 for testing. Actually, training MNIST is a problem of multiclass-classification since the data from MNIST is from 10 classes. According to [7], “one against one” strategy is used in such classification problems, so k(k-1)/2 classifiers are created where k is the number of classes. In MNIST, this means that 45 classifiers will be constructed before the training model is built.

We average the sum of all the running time in the training process and then do the prediction. The average time and the accuracy are listed in the table below and we find that the training time is reduced by 53.45% and the loss in prediction accuracy is about 0.08%. Therefore, the result is quite acceptable.

Table 3. Experiment on multi-class data sets

Data sets	Size of training examples	Size of test examples	Accuracy (LIBSVM /Parallel SVM)	Time Elapsed (LIBSVM / Parallel SVM) (ms)
MNIST	60000	10000	98.21% / 98.13%	15687.8 / 7302.178

6 Discussion

Based on the algorithm in [6, 7], we proposed a parallel method of training support vector machines. We choose and update both the maximal violating pair and the second maximal violating pair in the working set, approximately halving the time consumed by LIBSVM and maintaining the prediction accuracy in most cases.

However, in some cases the algorithm fails to converge. Although some of the problems are solved after tuning the parameter C and gamma, the convergence of the algorithm is not confirmed yet. In future works, we will attempt to make the algorithm surely converge by introducing the idea of function gain [12] into our work. After a considerable gain from the parallel step of our algorithm is obtained, we will stop the parallel phase of the algorithm and come back again to run LIBSVM which is serial so as to guarantee the convergence of our algorithm.

Acknowledgment. This work is supported by the National Natural Science Foundation of China via the Grant No. 60903092.

References

1. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152. ACM (July 1992)
2. Vapnik, V.: The nature of statistical learning theory. Springer (2000)
3. Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines (1998)
4. Osuna, E., Freund, R., Girosi, F.: An improved training algorithm for support vector machines. In: Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing VII, pp. 276–285. IEEE (September 1997)
5. Joachims, T.: Making large-scale support vector machine learning practical. In: Advances in Kernel Methods, pp. 169–184. MIT Press (February 1999)
6. Fan, R.E., Chen, P.H., Lin, C.J.: Working set selection using second order information for training support vector machines. The Journal of Machine Learning Research 6, 1889–1918 (2005)

7. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2(3), 27 (2011)
8. Cao, L.J., Keerthi, S.S., Ong, C.J., Zhang, J.Q., Periyathamby, U., Fu, X.J., Lee, H.P., Periyathamby, U., Fu, X.J., Lee, H.P.: Parallel sequential minimal optimization for the training of support vector machines. IEEE Transactions on Neural Networks 17(4), 1039–1049 (2006)
9. Graf, H.P., Cosatto, E., Bottou, L., Dourdanovic, I., Vapnik, V.: Parallel support vector machines: The cascade SVM. In: Advances in Neural Information Processing Systems, pp. 521–528 (2004)
10. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO algorithm for SVM classifier design. Neural Computation 13(3), 637–649 (2001)
11. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification (2003)
12. Glasmachers, T., Igel, C.: Maximum-gain working set selection for SVMs. The Journal of Machine Learning Research 7, 1437–1466 (2006)
13. LIBSVM–A library for Support Vector Machines, `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

Estimation of the Wetted Bulb Irrigation Drip through the Use of Neural Networks

Fabiana Costa de Araujo Schütz[1], Maria Hermínia Ferreira Tavares [2], Adriano de Andrade Bresolin[1], Eduardo Eyng[1], and Fernando Schütz[1]

[1] UFTPR - Technological Federal University of the Paraná, Brazil
Postal Box: 271, Av. Brasil, 4232
CEP 85.884-000, Medianeira, PR, Brazil
{fabianaschutz,aabresolin,eduardoeyng,fernando}@utfpr.edu.br
[2] UNIOESTE - State University of Western Paraná
Rua Universitária, 2069. Bairro: Jardim Universitário
CEP 85819-110, Cascavel, PR, Brazil
maria.tavares@unioeste.br

Abstract. This research aimed to program and test a neural network can predict the size of the wetted in drip irrigation from about pressure and flow. The study is important for the design of drip irrigation systems that aim to minimize costs and avoid waste in the application of water, fertilizers and other inputs. The neural network model was used with the supervised back propagation algorithm. Various network topologies with different activation functions were tested, and one presented the best index prediction, ie, lower error was found with five layers being 4:20:40:20:2 with linear activation function. The cross validation system was used to validate the results statistically, since the number of measurements is small (120). The final average error for the bulb diameter was at 6.67%, while the average error to end the vertical distance resulted in 9.09%.

Keywords: drip irrigation, wetted bulb, neural networks.

1 Introduction

The rational management of water in irrigation ensures the best use of a smaller amount of water, at a lower cost. Knowledge of the soil's ability to store water in irrigated crops is crucial for the proper establishment of the frequency of irrigation.

To this end it created the precision irrigation, which consists in monitoring the area to be irrigated, providing up water only where and when appropriate, thereby avoiding wastage and preventing water stress in plants. In drip irrigation the best way to know the distribution and, consequently, the redistribution of water in soil is by monitoring the wetted bulb.

For this purpose, it is necessary to develop tools that allow the monitoring of water supply for each soil type, as the distribution and redistribution of soil water are different according to the characteristics of the soil. One of the tools used for this

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 665–672, 2013.

analysis is the modeling, able to simulate various combinations of working pressure, flow rate and dimensions of the wetted.

In the paper of Levin et. al. [1] was estimated size of wet soil volume in different soils under surface drip irrigation, using mathematical models in the literature. As a result it was concluded that the simulated data volume dimensions of wet soil of medium texture, for different flow rates of emitters at different times of application, can be used to guide irrigation in similar soils, noting that the models numerical and analytical exhibit more reliable results [1].

In this context, the present study aimed to program and test a neural network can predict the size of the wetted from information of pressure and flow. The study is important for the design of drip irrigation methods that seek to minimize costs and waste mainly in the application of water, fertilizers and other inputs.

2 The Importance of Precision Irrigation

The method of drip irrigation is in one of the best alternatives to overcome the occurrence of significant spatial variability of physical, chemical and morphological characteristics of soils in irrigated areas. Through simple operational modifications, the method can take into account possible variations in retention characteristics and movement of water in irrigated area. In drip irrigation operating pressures are generally low and small changes in pressure produce significant effect on the variation of flow and, consequently, the uniformity of water application, increasing the accuracy of the application.

At paper Burneya et. al. [2], the researchers assessed the drip irrigation powered by solar energy as a strategy to improve food security in rural Sudano-Sahelian West Africa. They observed that the drip irrigation solar-powered household income increased significantly compared to alternative technologies. Moreover, they also found improvements in the three components that ensure food security, increasing the steady supply of food, significant improvement in the ability of nutritional and still increase the ability to consume and benefit from nutritious foods, showing the positive impacts of drip irrigation in the region studied [2].

The main way to study the efficiency of drip irrigation is through the evaluation of the wetted bulb. The shape and size of the bulb depend on the flow rate applied, the type of transmitter, the duration of irrigation and the soil type.

3 Neural Networks

Neural networks are models of inference and nonlinear multidimensional [3]. The great advantage of these models is their ability to "learn", generalizing rules automatically extract or complex datasets. Currently, neural networks are being applied successfully in a wide range of problem domains, in areas such as finance, medicine, engineering, geology and physics [4].

As a neural network can accept different input data, the use of this technology proved to be adequate for modeling data collected in the field, such as: data monitoring water quality, hydrological parameters for disaster prediction, topographic

information, edaphic parameters, biomass values, stages of crop development, simulaçõesda influence of environmental conditions on the water quality and the flow of small water bodies, soil water distribution, among others.

At paper Sporl, Castro and Luchiari [5], the researchers developed a methodology for building models of environmental fragility in neural networks and concluded that the RNA is efficient to do this translation. The results showed that it is possible to emulate, with reasonable reliability, the standard of review in the definition of weak environmental systems [5].

The behavior of discriminant variables in digital soil mapping using artificial neural networks was evaluated in the work of Chagas et. al. [6]. The maps generated by each of the six sets of variables were compared to reference points for determining the accuracy of classification. It was concluded that the approach using neural networks can help make the survey solosmais faster and less subjective [6].

The efficiency of artificial neural networks, compared with the maximum likelihood classifier for the classification of land use, with emphasis on levels of pasture degradation in City of Viçosa, State Minas Gerais - Brazil, has been studied by other work Chagas et. al. [7]. In this work, it was observed that the use of artificial neural networks for classification of the use and land cover is extremely efficient.

The work Kisi [8] investigated the potential of techniques and MLP RBNN (Recurrent Backpropagation Neural Network) for application in the investigation of monthly estimates of evaporation using climatic data entries. It was found that the models used are more suitable than other applications in multi-stations and that such techniques ANN (Artificial Neural Networks) can be applied in the modeling of water use in project reservoirs, and several other hydrological applications [8].

4 Methodology Applied

The experiment was conducted in a greenhouse built in the Experimental Center of Agricultural Engineering of the State University of West Paraná, city Cascavel, State of Paraná-Brazil, whose geographical coordinates are: latitude 24 ° 53'S, 53 ° 23'W longitude and altitude of 682 meters. The soil of the region is classified as Hapludox [9]. The soil of the experiment consists of 68% clay, 13% silt and 19% sand.

To measure the dimensions of the wetted bulb inside the greenhouse, a trench of 1 m3 was opened, across the irrigation line. It was installed a drip pipe with a flow rate of 3.73 l / h / m and pressure ranging from 3, 5, 7 and 9 mca with emitter spacing of 20.3 cm. The slope was found in level, working with a uniformity of distribution of 95% and for that, lines with 61 m long were used. Two pressure gauges were installed at the beginning and at the end of the irrigation line to control the pressure along the irrigation. In the field wet bulb was measured at intervals of 30, 60, 120, 180 and 240 minutes, being used later when the results will be compared with the simulated dimensions through the neural networks model.

Altogether 120 repetitions were performed, so in order to make the model it was developed a neural network composed of multilayers with a backpropagation algorithm. In it, the learning is done based on the error included in the response provided by the network, which means that the difference between the actual response

of the simulation and the desired one should be minimized, being needed therefore the adjustment of the synaptic weights associated with the inputs of neurons.

The input layer has 4 parameters: pressure, flow, time and volume of water. In the output layer, two neurons are responsible for processing the responses of the network, which are the dimensions of the wetted bulb and the vertical distance. As for the hidden layers, they were determined with several simulations, seeking good results with the least possible computational effort. In the validation phase, we analyzed the behavior of the network for a number of different values from the one used in the previous phase, using cross-validation.

5 Experiments and Results

In Figures 1, 2, 3, and 4 it is possible to observe the dimensions of bulbs measured in the field over the time (30, 60, 90, 120 and 240 minutes) of the application to the pressures of 3, 5, 7, and 9 mca respectively.

It can be seen that for the first hour of irrigation water distribution in the wet bulb was greater in the horizontal direction and in the second hour of irrigation, there is a greater vertical distribution than horizontal, probably due to the volumetric moisture in this period of irrigation being already very close to 50%, favoring the redistribution of water where the soil is near saturation humidity. However, from the third hour to the fourth hour of irrigation, there was no change in the value of the vertical dimension, which indicates that redistribution is now again more pronounced in the horizontal direction.

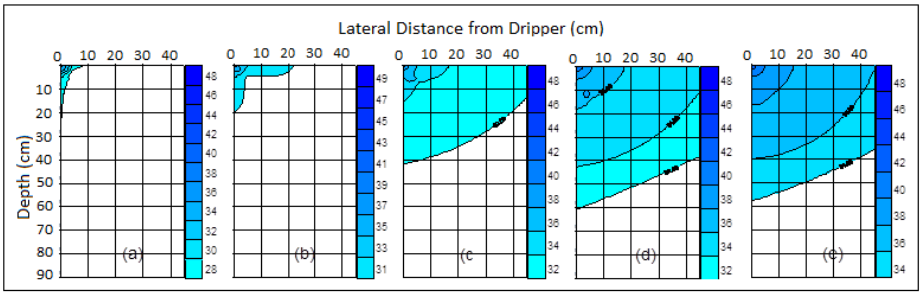


Fig. 1. Dimensions of the wetted irrigation over time with working pressure of 3 mca

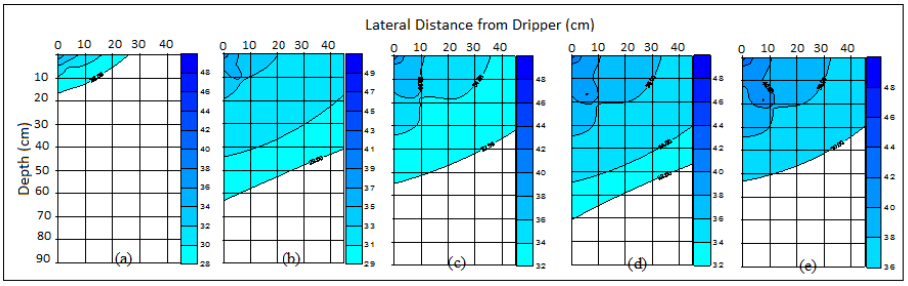


Fig. 2. Dimensions of the wetted irrigation over time with working pressure of 5 mca

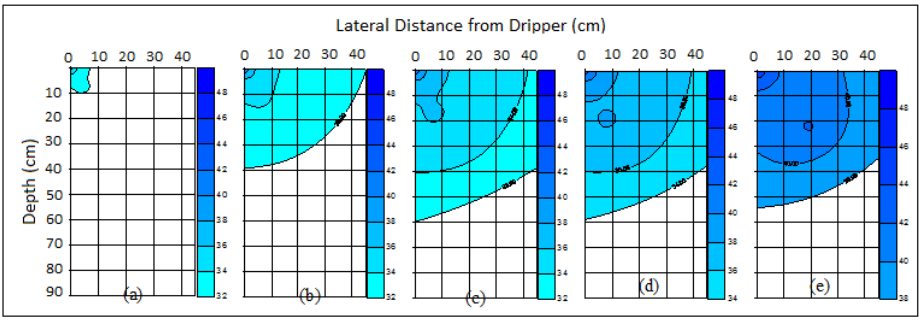


Fig. 3. Dimensions of the wetted irrigation over time with working pressure of 7 mca

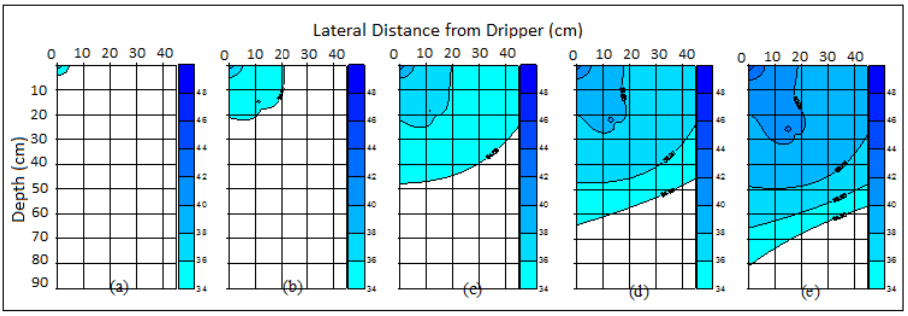


Fig. 4. Dimensions of the wetted irrigation over time with working pressure of 9 mca

To better evaluate the validity of the neural network to estimate the size of the bulb, the data collected were divided into 5 experimental random sets of 24 elements each (A, B, C, D, and E). The cross validation combinations are shown in Table 1.

Table 1. Combinations tested in cross-validation

Experiment	Training Set	Validation Set
Training Test 01	A, B, C, D	E
Training Test 02	B, C, D, E	A
Training Test 03	C, D, E, A	B
Training Test 04	D, E, A, B	C
Training Test 05	E, A, B, C	D

Initially we chose a network with three layers and number of neurons 4:20:2. This network showed an error rate above 30%. Testing it in a larger network with 4:100:2, the average error was above 30% again. Then the size of the network was increased to 4 layers. Again two tests were performed with two configurations 4:20:20: 2 and 4:40:40:2. In both cases, the average error improved but still above 17%. In all previous tests were tested linear activation function, sigmoid and hyperbolic tangent, and the best results were obtained with linear activation function.

Table 2 shows the results obtained by cross-validation set for A to E, to the network with five layers 4:20:40:20:2 with linear activation function. The training and validation process took about 21 hours. The final average error for the diameter of the bulb was at 6.67%, while the final average error in the vertical distance was 9.09%.

Table 2. Results obtained by cross-validation

Experiment - (Set)	Average Error (%) Diameter Bulb	Average Error (%) Vertical Distance
Validation Test 01 - (E)	6.76	9.87
Validation Test 02 - (A)	6.53	8.75
Validation Test 03 - (B)	6.97	7.44
Validation Test 04 - (C)	7.32	9.96
Validation Test 05 - (D)	5.78	9.41

The Figures 5, 6, 7, 8 and 9 show the results in the comparison between measured data and estimated data for the neural network.

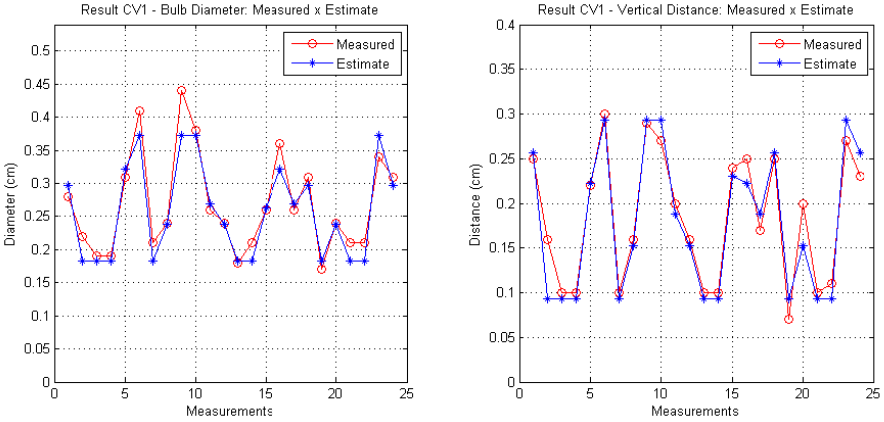


Fig. 5. Results of Cross Validation - Test 01: Neural Network Estimate x Measured values

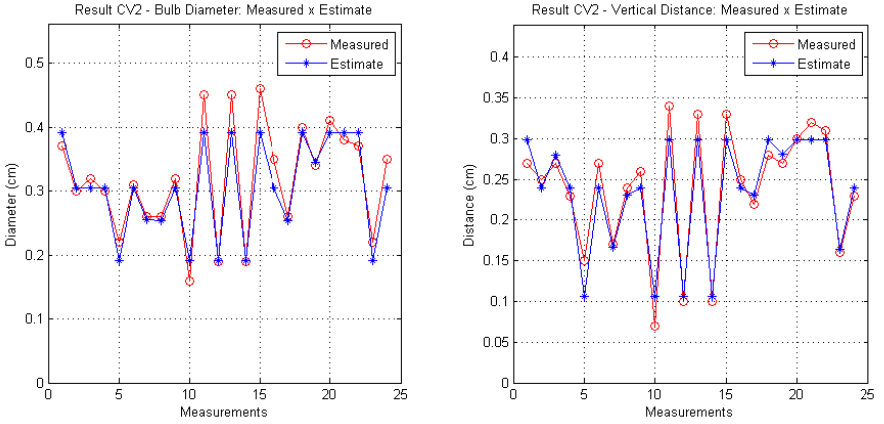


Fig. 6. Results of Cross Validation - Test 02: Neural Network Estimate x Measured values

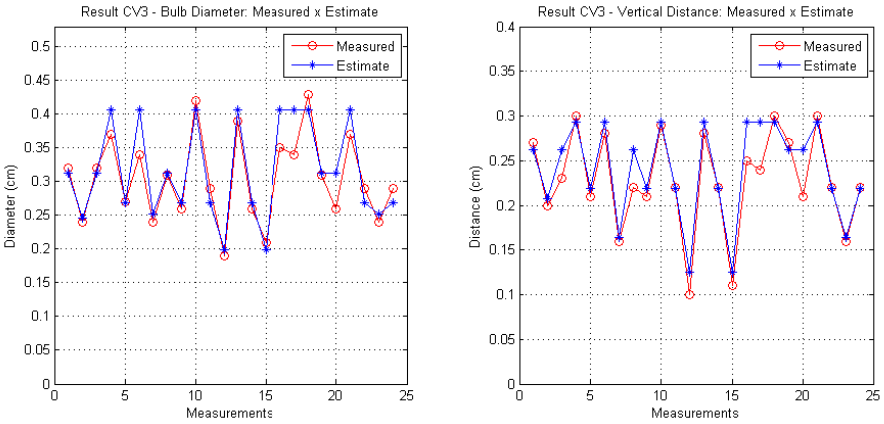


Fig. 7. Results of Cross Validation - Test 03: Neural Network Estimate x Measured values

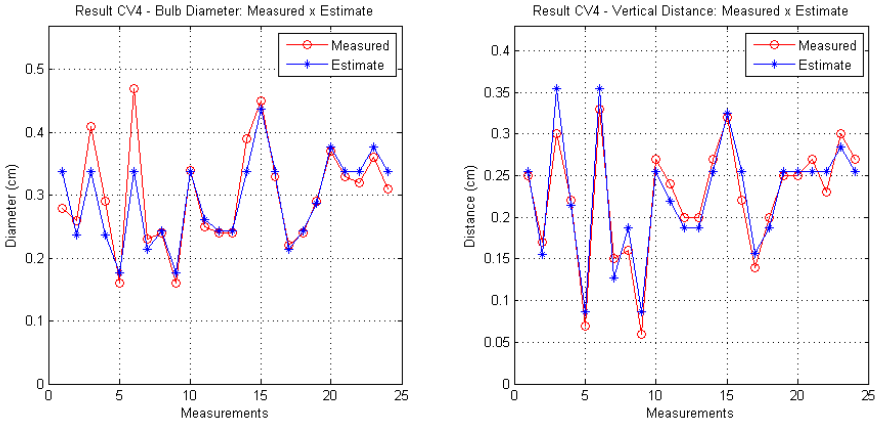


Fig. 8. Results of Cross Validation - Test 04: Neural Network Estimate x Measured values

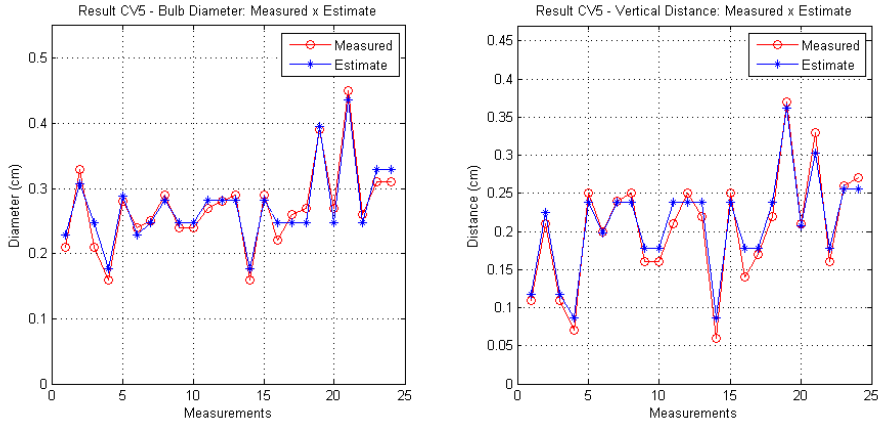


Fig. 9. Results of Cross Validation - Test 05: Neural Network Estimate x Measured values

Several neural network configurations were used, the criterion was to evaluate configurations close to that which presented the best fit.

To compare the variances of the measured and estimated values for the network validation sets F-test was applied, which is one of the tools of data analysis, which involves comparing the results to variations in population. The result data are presented in Table 3.

Table 3. Results obtained by F-test

Experiment	p-value	Confidence interval for the ratio of the variance
Validation 01	0.84	99%
Validation 02	1.26	99%
Validation 03	0.76	99%
Validation 04	0.97	99%
Validation 05	1.36	99%

Through table 02, it can be concluded that the estimates made by the neural model for the five data sets evaluated had their variances equal to the measured data, the 99% confidence level. Finaly, it can be concluded, therefore, that the modeling of the distribution of ground water using RNA satisfactorily meet the application tested, since it allows the characterization of the wetted times and to certain crops, favoring the application of water with precision while minimizing waste of water.

References

1. Levien, S.L.A., Miranda, J.H., Bezerra, A.H.F.: Estimating dimensions of volume of wet soil surface drip irrigation on medium textured soil: initial condition of dry soil. Brazilian Journal of Irrigated Agriculture 6(2), 127–135 (2012)
2. Burneya, J., Woltering, B.L., Burkec, M., Naylora, R., Pasternak, B.D.: Solar-powered drip irrigation enhances food security in the Sudano–Sahel. Proceedings of the National Academy of Sciences of the United States 107(5), 1848(6) (2010)
3. Haykin, S.: Redes Neurais: Princípios e prática. In: Neural Networks: Principles and Practice, 2nd edn. Editora Bookman, Porto Alegre (2005)
4. Boyukata, M., Kocyigit, Y., Guvenc, Z.B.: Estimation of Cross Sections for Molecule-Cluster Interactions by Using Artificial Neural Networks. Brazilian Journal of Physics (2006)
5. Spörl, C., Castro, E.G., Luchiari, A.: Aplicação de Redes Neurais Artificiais na Construção de Modelos de Fragilidade Ambiental. Revista do Departamento de Geografia – USP, Brazil 21, 113–135 (2011)
6. Chagas, C.S., et al.: Topographic attributes and data Landsat7 in digital soil mapping using neural networks. Pesq. Agropec. Bras. 45(5), 497–507 (2010)
7. Chagas, C.S., Vieira, C.A.O., Fernades Filho, E.I.: Use of artificial neural networks in the classification of levels of pasture degradation. Revista Brasileira de Engenharia Agrícola e Ambiental – Agriamb 13(3) (2009)
8. Kisi, Ö.: Modeling monthly evaporation using two different neural computing techniques. Irriga Science 27, 417–430 (2009)
9. Embrapa - Centro Nacional de Pesquisa de Solos (National Research Center of Soil - Rio de Janeiro, RJ). Brazilian System of Soil Classification - Brasília: Embrapa Produção de Informação, p. 412 (1999)
10. Oliveira, M.V.A., Villas Boas, R.L.: Uniform distribution of potassium and nitrogen in drip irrigation system. Eng. Agríc. 28(1), 95–103 (2008)

Investigation of the Predictability of Steel Manufacturer Stock Price Movements Using Particle Swarm Optimisation

Pascal Khoury[1,2] and Denise Gorse[1]

[1] Dept of Computer Science, University College London,
Gower Street, London WC1E 6BT, UK
[2] Charlemagne Capital, 39 St. James's St., London SW1A 1JD, UK
{P.Khoury,D.Gorse}@cs.ucl.ac.uk

Abstract. It is shown that ensemble classifiers composed of neural networks trained using particle swarm optimisation can uncover a substantial degree of predictability in stock price movements. As in a previous work by the authors use is made here of a training metric, the Matthews correlation coefficient, that has been shown to better handle numerically unbalanced data sets. The work provides a solid basis for the future construction of a trading model.

Keywords: Particle swarm optimisation, ensemble classifiers, forecasting, portfolio management.

1 Introduction

In spite of past academic scepticism as to the predictability of markets [1] there do appear to be periods of predictability [2]. Investors are demanding of active money management that can exploit such opportunities. However managers need to be wary of 'black box' solutions, since some patterns may persist only for short time periods and it is important to pre-empt changes in underlying market structure by retiring from use inappropriate models. Thus the ideal model from a fund manager's point of view would be one that both facilitates profitable decisions and also to some degree explanation for these decisions. This paper presents results that while preliminary go some considerable way toward achieving these simultaneous desirable objectives.

Steel companies were used in this study because they have been among the worst performers since the global financial crisis in 2008. Although a recovery is not likely to be imminent, it is nevertheless perceived to be possible beyond 2013, the argument being supported by the fact that the price of steel is currently at its lowest since July 2009. This study aims to highlight some of the factors that drive stock prices for steel companies, and tests the ability of these drivers to predict aggregate returns using a methodology based on previous work by the authors [3], which showed that an ensemble of neural networks trained using particle swarm optimisation (PSO), and making use of the Matthews correlation coefficient as a fitness function, could identify those financial factors most strongly associated with firm-specific risk.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 673–680, 2013.

It will be shown that as in our earlier work on volatility and risk [6] not all input factors are equally important, but here that in addition the relative value of these factors (here as predictors of profit) changes over time. It will further be shown that average weight vectors for voting members of the neural network ensemble can be clustered so as to allow a user of the system to make a judgement, in advance of its use, as to the reliability of the profitability predictions that will be made. This last would appear to be a valuable tool for investment decisions and risk management.

2 Data Set

The data used were from 307 steel companies, gathered from the first quarter of 1995 to the first quarter of 2012, with the objective of trying to use a set of 32 factors representing a snapshot over a quarter year of a company's 'financial health' to predict positive or negative returns for that company both over the next month and next year. All the factors are considered by the investment industry to be at least potentially predictive of market behaviour; we however find certain of them to be considerably more important than others, those which were found to be most strongly predictive in the context of the present problem being discussed in more detail in Section 4.2. Each prediction task comprised a set of 69 separate sub-tasks, created by progressively moving by three months a training window corresponding to the previous quarter's factor measurements. The return prediction problem was here posed in binary form, dividing instances into those corresponding to later profit or loss, but predicting the degree of this could become the subject of later work.

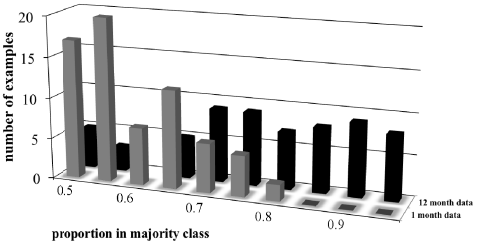


Fig. 1. Bar chart illustrating number of examples (time periods 1..69) in relation to the proportion of stock prices in the majority (either profit or loss) class at that time

For each of the 69 time periods, for each task, the 307 companies were divided into the profit (target 1) or loss (target 0) category. For many of the periods this is far from being a 50/50 split, as is shown in Fig. 1, with strongly unbalanced data sets being especially common for twelve month forward prediction. In situations like this there is often a tendency to assign all examples to the majority class. For this reason we as in [3] use the Matthews correlation coefficient (MCC), a counting measure defined in Section 3.2 that strongly penalises over assignment to a majority class, as a training

fitness measure, since the MCC was shown in our previous work to be more effective than training on mean squared error for unbalanced data sets.

The data were chosen with a focus on identifying factors that could be used to predict price movements, not on the back-testing of a trading model, and for this reason factor averages were taken over the twelve weeks comprising each quarter of the moving window, and these values used to predict the signs of forward returns. However the relevant factors that were identified could of course become the basis for a trading model, and it is intended to make this the subject of future work.

3 Methods

3.1 Particle Swarm Optimisation

Particle swarm optimisation (PSO) [4] is a biologically inspired, population based search algorithm that has been applied successfully in many areas, including finance [3,5,6]. Every particle in the swarm has a velocity $\mathbf{v}_i$ and position $\mathbf{x}_i$, where the latter will here correspond to the full list of weights possessed by the ith neural network. The equations used to update the velocity and position are

$$\mathbf{v}_{i,t+1} = W\mathbf{v}_{i,t} + \phi_1\beta_1(\mathbf{p}_{i,t} - \mathbf{x}_{i,t}) + \phi_2\beta_2(\mathbf{g}_t - \mathbf{x}_{i,t}) \quad , \tag{1a}$$

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \mathbf{v}_{i,t+1} \tag{1b}$$

where $\mathbf{p}_{i,t}$ is the best position (weight set) found at time t by particle (net) i, $\mathbf{g}_t$ (*gbest*) is the best position found at this time by any particle, β_1, β_2 are random numbers chosen uniformly from the interval [0,1], and W is an iteration-decreasing *inertia weight*. In the current work as in [3] 500 training iterations were used, with a swarm size of 25 particles, $\phi_1 = \phi_2 = 2$, with decreasing inertia weight W in the range [1, 0.2].

3.2 Matthews Correlation Coefficient (MCC) as a Fitness Metric

The MCC is for a two-class problem defined by

$$MCC = \frac{n_{11}n_{00} - n_{01}n_{10}}{\sqrt{(n_{11}+n_{01})(n_{11}+n_{10})(n_{00}+n_{01})(n_{00}+n_{10})}} \, , \tag{2}$$

where n_{00} (n_{10}) is the number of true (false) negative examples and n_{11} (n_{01}) is the number of true (false) positive examples. An MCC of 1 denotes perfect performance, a value of 0 either a random guess or the assignment of everything to one class. Because the MCC is a counting measure its use as a training metric is problematic for methods such as error backpropagation that require a differentiable performance metric; differentiability of the fitness function is not however necessary for PSO, and this feature of PSO is one of the motivations for its use in the current work.

3.3 k-Means Clustering

This was required for the weight vector analysis of Section 4.2, and was done using Matlab. The Euclidian distance was minimised to estimate the cluster centroids, which were initially approximated by partitioning 10% of the data randomly.

3.4 Training Methodology and Assessment of Results

Each of the 69 consecutive experiments consisted of five runs, each run consisting of:

- The set of 307 companies is randomised and divided into five non-overlapping but roughly equally sized subsets which will be the five test sets for this run;
- For each such division the remaining four-fifths of the data is itself divided into five non-overlapping subsets and fivefold validation carried out, with the *gbest*s at each iteration also assessed on the validation data, the *gbest* weight set saved from each training/validation split being that which did best on the validation set;
- The five weight sets thus selected are used as a committee to classify the test set by taking an average of their outputs.

As in our previous work [3] it was discovered here that using multilayer nets as swarm/ensemble members did not improve classification performance on validation sets; a not unexpected result given the amount of noise typically present in financial data, but as in [3] useful in that with single layer nets the magnitudes of input layer weights can then give an indication of the degree to which factors are involved in the classifier decision process. Test results (each of the 307 companies during a given run being used for testing once and only once) were assessed according to the *normalised percentage better than random* (NPBR), defined here as in [3].

4 Results

4.1 Predictability of One Month and Twelve Month Returns

This section will investigate the degree to which aggregate returns (averaging across all 307 steel companies) can be predicted from the 32 financial factors mentioned in the introduction, highlighting those factors that appear most predictive over the one month and twelve month periods, and showing that restriction of the input factors to only those deemed most predictive can improve the reliability of the predictions.

Fig. 2 below shows the average (± one standard deviation) NPBR achieved for one month or twelve month forward prediction for all 69 successive financial quarters (around six years of data), using the training methodology described above. The breaks in the NPBR curves for the twelve month problem correspond to periods in which data are too severely imbalanced (for example in time period 36 *all* stock price returns were positive) to support training, even using the MCC as a fitness measure.

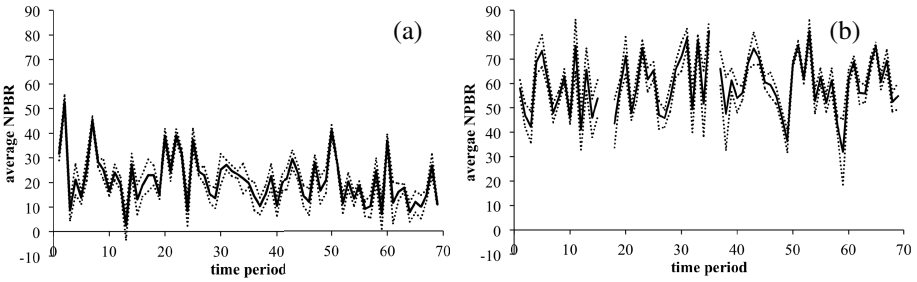


Fig. 2. Prediction quality in terms of normalised percentage better than random (NPBR) as a function of time period for (a) one month, and (b) twelve month, data

It is immediately apparent from the above that there appears to be a high degree of predictability in the twelve month returns (Fig. 2b), and a substantial degree of predictability also in the one month data (Fig. 2a). The difference in predictability of monthly and yearly returns is due to the fact that yearly returns are likely to contain less noise than monthly ones, and, more specific to the steel industry, that the price of iron ore—the raw material used by steel manufacturers—has been moving in recurring cycles of nine to twelve months, this cyclic behaviour making the twelve month share price performance somewhat more predictable.

It should be emphasised that while the results of Fig. 2 are very encouraging they may not turn automatically into equally striking profits. Transaction costs, frequently underplayed in academic forecasting studies, need to be considered. In addition from a fund manager's standpoint it is important to minimise drawdowns as a portfolio that achieves a large profit while en route displaying erratic performance will be unacceptable to many clients. The related idea that not all recommended changes of position should, in the presence of transaction costs, be acted upon, is one that will be returned to in Section 4.3, when it is shown it may be possible to pre-judge the reliability of an ensemble predictor and avoid its use at less favourable times, a feature we expect to make use of in future work implementing a full trading model.

4.2 Identification of Significant Input Factors

In our previous work [3], which used ensemble predictors to identify financial factors characterising volatility and firm-specific risk, it was shown that the average magnitude of a weight across all voting members of the classifier could be used to determine the predictive value of the associated input variable more reliably than single-variable input-output correlation, and that the model in this case benefited from the elimination of factors associated with small-valued weights.

The predictability of price movements was therefore re-investigated with the elimination, at each time period, of those input variables that in the original experiments, for that time period, had a magnitude of less than 2, with comparative results (against the use of the full 32 variables) shown in Fig. 3 below.

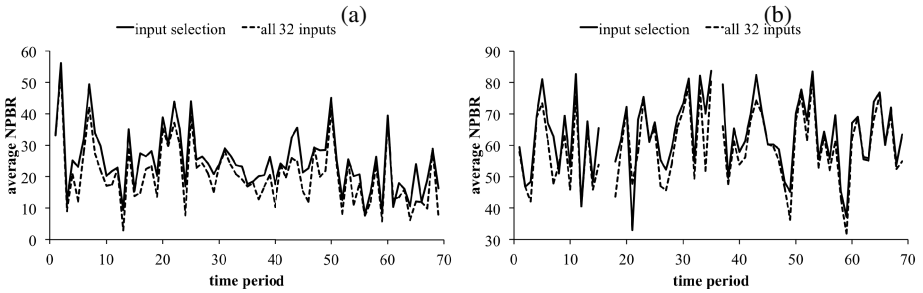


Fig. 3. Comparison (in terms of average NPBR) of results using all 32 input factors with factor subsets based on weight magnitudes for a) one month, and (b) twelve month, data

It is clear from this that the use of a restricted subset of input features (on average 13/32 of them being used), with the choice based on the magnitude of the average weights for that position, does not damage the system's ability to predict price movements, and for the one month case significantly improves it (with $p < 0.005$).

The bar charts of Fig. 4 below give some idea of the usage of specific input factors by showing the number of times a weight has the largest positive (tending to predict a positive return) or largest negative (tending to predict a negative return) weight over the complete experimental period of 69 financial quarters.

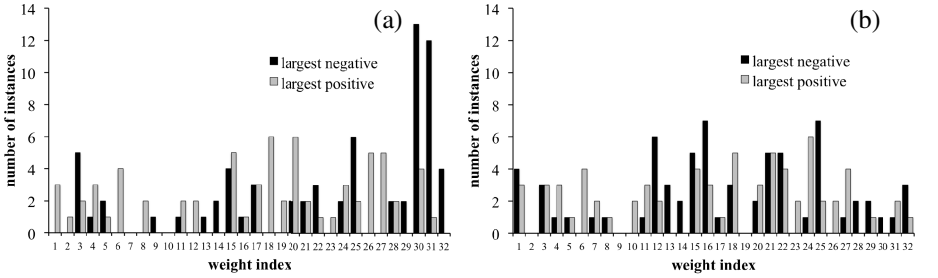


Fig. 4. Numbers of times during the 69 prediction periods when ensemble-averaged weights have the largest positive or negative values for (a) one month, and (b) twelve month, data

In the one month case the high negative weighting on inputs 30 and 31 reflects the expected effect of price reversal. Input 18 (positive earnings revisions), is also interesting: as might be expected it appears to promote positive one month forward returns. On the other hand, the high weighting on inputs 26 and 27 (respectively the company's total assets and total shareholder equity) is surprising as we would not have expected those to be strong indicators at the outset. This latter result will prompt further investigation. In the case of the twelve month prediction two factors are particularly notable: first, the five year growth in sales (positive input 24) emphasizes the importance as perceived by the market of long-term growth in sales for steel manufacturers; second, the strong negative weighting placed on the return on invested capital (input 12), which we find an interesting result and worthy of further inquiry.

4.3 Clustering Analysis of Ensemble Weight Vectors

It is clear from looking at Figs. 2 and 3 that forward returns can be predicted considerably more effectively at some times than others. Bearing in mind that a small number of large losses can outweigh the benefit of many small gains, and that incorrect predictions, in reality and within a realistic trading model, carry a double penalty because of transactions costs, it would be of obvious benefit to be able to determine, when considering acting on a set of predictions for the next month or year's company performances, whether such predictions could be trusted.

It was decided to investigate whether a k-means clustering analysis, performed on weight vectors which were as for Section 4.2 averaged over all voting members of the ensemble, could identify useful characteristics that could serve to distinguish, in advance of their use, trustworthy versus untrustworthy predictor systems.

Values of k=2, 3, and 4 were used in the clustering analysis, with five runs of the process for each value of k, with the clearest criteria for clustering emerging for k=2. These criteria were discovered by averaging the weight vectors corresponding to the two centroids for each run (it was clear which was which in each instance because the results of the five runs did not differ very greatly) to create prototype weight vectors, and re-clustering the averaged weight vectors for each of the 69 time periods with respect to these. The results are summarised in the table below, with the forms of the associate prototype weight vectors shown in Fig. 5.

Table 1. Average confidence, as measured by NPBR (1) and proportion of predictions of positive returns (2) for ensemble weight vectors reclustered in relation to the prototypes displayed in Fig. 5 for one month and twelve month data. The dominant clustering criteria (either 1 or 2) are in each case highlighted.

	1 month data		**12 month data**	
	cluster 1	cluster 2	cluster 1	cluster 2
1: <NPBR>	**24.68±9.37**	**18.87±9.40**	59.54±11.05	54.96±18.46
2: <prop. predictions of positive returns>	0.51±0.15	0.50±0.15	**0.70±0.26**	**0.39±0.29**

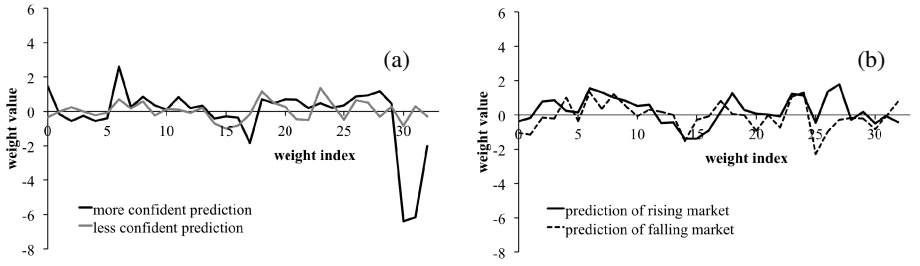


Fig. 5. Run-averaged weight vector centroids for k=2 clustering and their associated interpretations for (a) one month, and (b) twelve month, data

As can be seen from the table, for the one month forward return problem the primary splitting of ensemble classifier weight vectors does indeed appear to be on the basis of the classifier's reliability, measured by the average NBPR achieved during those time periods belonging to the relevant cluster. For the twelve month forward return problem, which is somewhat easier, the primary basis for clustering was instead the proportion of predictions of positive returns, essentially a prediction of whether the steel market as a whole will rise or fall during the next year. It can further be noted that while k=3 clustering was somewhat difficult to interpret, k=4 clustering created a split in each of the clusters discovered at the k=2 level into subclusters based on whichever of the above two criteria was not used at the primary k=2 level.

5 Discussion

The work presented here has established it is possible to discover financial factors relevant to the prediction of returns in the steel industry, one for which predictions are considered to be challenging. Results for the prediction of positive versus negative returns were sufficiently above random; we believe the incorporation of trading costs will not undermine the system, and we now intend to create a realistic trading model for backtesting. Within this trading model we believe it may be possible to use the results of our clustering analysis, in particular for k=2, to create weight filters based on past training experience that will allow an informed decision to be made about the use of a newly trained predictor. Overall we feel our results are encouraging, and while the use made by the PSO ensemble of some of the financial factors was found at least at first inspection to be counter to industry intuition, the quality of the results strongly supports further work in the direction of the construction of a trading model.

References

1. Grinold, R.C., Kahn, R.N.: Active Portfolio Management. McGraw-Hill, NY (1999)
2. Litterman, B.: Modern Investment Management. Wiley, NJ (2003)
3. Khoury, P., Gorse, D.: Identification of Factors Characterising Volatility and Firm-Specific Risk Using Ensemble Classifiers. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part IV. LNCS, vol. 7666, pp. 450–457. Springer, Heidelberg (2012)
4. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: IEEE International Conference Symposium on Neural Networks, pp. 1942–1948. IEEE Press, New York (1995)
5. Poli, R.: An Analysis of Publications on Particle Swarm Optimisation Application. Technical report, Department of Computer Science, University of Essex (2007)
6. Banks, A., Vincent, J., Anyakoha, C.: An Review of Particle Swarm Optimization. Part II: Hybridisation, Combinatorial, Multicriteria and Constrained Optimization, and Indicative Applications. Natural Computing 7, 109–124 (2008)
7. Dunis, C., Laws, J., Naïm, P.: Applied Quantitative Methods for Trading and Investment. Wiley, NJ (2003)

SVM Analysis of Haemophilia A by Using Protein Structure

Kenji Aoki[1], Kunihito Yamamori[2], Makoto Sakamoto[2], and Hiroshi Furutani[2]

[1] Information Technology Center, University of Miyazaki,
Miyazaki City, Miyazaki 889-2192, Japan
aoki@cc.miyazaki-u.ac.jp
[2] Faculty of Engineering, University of Miyazaki,
Miyazaki City, Miyazaki 889-2192 Japan
{yamamori,sakamoto,furutani}@cs.miyazaki-u.ac.jp
http://www.miyazaki-u.ac.jp/

Abstract. Haemophilia A is a genetic disease resulting from deficiency of factor VIII. The database of mutations causing haemophilia A has been developed by the world wide collaboration. In this study, we examined the relation between activity of factor VIII and the missense mutation by using Support Vector Machine (SVM). As parameters, we used four physical-chemical parameters of amino acids and a structural feature. As a result, we predicted the severity of haemophilia A by using SVM in A domains of factor VIII. The structural parameters influence the prediction in A domains.

Keywords: Haemophilia A, Factor VIII, Amino Acid, Support Vector Machine.

1 Introduction

The haemophilia is a group of hereditary genetic disorders, in which one of the coagulation factors is deficient [1]. Haemophilia A is the most common form of disorder caused by low concentration of the coagulation factor VIII. Haemophilia B is another form of disorder caused by deficient factor IX. Haempphilia A accounts for about 85% of this disorder, while haemophilia B for $10 - 12$ % [2]. Haemophilias A and B are clinically indistinguishable from each other. Diagnosis must be confirmed by specific factor assay.

It becomes very important to study mutations in genes responsible for diseases by biological experiment. However, it is a time-consuming, laborious and expensive task. Thus, it is necessary to develop computational method by applying various approaches. We used a multiple regression model to predict the effect of a missense mutation in Factor IX gene of haemoplilia B patients [3]. In ICONIP-2012, we have demonstrated the calculations using Support Vector Machin (SVM) for the analysis of mutant Factor IX genes [4].

There have been reported a variety of defects in the factor VIII gene from haemophilia A patients[5], and these are summarized in the haemophilia A

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 681–688, 2013.

database [6]. In this study, we analyzed amino acid changing mitations, or missense mutations in the database described with factor VIII activity values. We adopted 439 cases from the database. We use the distances between 20 amino acids by using the four physical-chemical properties: Molecular volume, Hydropathy, Polar requirement and Isoelectric point.

In this study, we also use the parameters obtained by using the 3D structure of factor VIII protein; Accessible area and Fractional sidechain solvent accessibility. The values of these two parameters are presented in the database [6].

2 Haemophilia A

The gene coding for human factor VIII consists of 26 exons and 25 introns, and is located on the X chromosome [5]. Factor VIII is an essential blood-clotting protein, and synthesized as a precursor protein of 2351 amino acids. This includes a signal peptide and a mature protein of 2332 amino acids with domain structure A1-A2-B-A3-C1-C2. Three A domains display approximately 30% homology to each other. The C domains are structurally related to the C domains of factor V. The B domain exhibits no significant homology with any other known protein.

We used Haemophilia A Mutation Database [6].

Table 1. Domain Structure of Factor VIII

Domain	Location	Number of mutants
A1	1 to 329	111
A2	330 to 711	131
B	712 to 1648	18
A3	1649 to 2019	107
C1	2020 to 2172	39
C2	2173 to 2332	33

Activity of factor VIII in a patient's blood depends on a position of the substitution and combination of original and substituting amino acids. Classification of haemophilia A is presented in Table 2.

3 Methods

3.1 Support Vector Machine

Support vector machine (SVM)[8][9] can classify the samples $\boldsymbol{x}_i$ $(i = 1, \cdots, n)$ belonging to unknown class into two classes C_1 or C_2. The classification function $f(\boldsymbol{x})$ is defined as the Equation (1).

$$f(\boldsymbol{x}) = sign\,(g(\boldsymbol{x})) = sign(\boldsymbol{w}^t\boldsymbol{x} + b), \tag{1}$$

where $\boldsymbol{w}$ and b are parameters.

Table 2. Classification of Haemophilia A

Concentration	Classification	Clinical
$< 1\%$ of normal	Severe	Spontaneous joint and muscle bleeding, bleeding after injuries, accidents, and surgery
$1 - 5\%$ of normal	Moderate	Bleeding into joints and muscles after minor injuries, excessive bleeding after surgery
$5 - 40\%$ of normal	Mild	Spontaneous bleeding does not occur, bleeding after surgery, and accidents

Let $\boldsymbol{x}_i$ belong to the class y_i $(= \{1, -1\})$, and if all the samples are correctly classified, Equation (2) will be satisfied.

$$\forall i,\ y_i \cdot \left(\boldsymbol{w}^t \boldsymbol{x}_i + b\right) - 1 \geq 0. \tag{2}$$

When Equation (2) is satisfied, no samples exist between the $H_1 : (\boldsymbol{w}^t \boldsymbol{x} + b) = 1$ and the $H_2 : (\boldsymbol{w}^t \boldsymbol{x} + b) = -1$, and the distance between H_1 and H_2, called as *margin*, becomes $\frac{2}{||\boldsymbol{w}||}$. To obtain the maximum margin, we minimize $\frac{1}{2}||\boldsymbol{w}||^2$. In SVM, it is solved by a Lagrange-multiplier method. To maximize the margin, we rewrite the objective function as Equation (3) in subject to Equation (2),

$$L\left(\boldsymbol{w}, b, \boldsymbol{\alpha}\right) = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{n} \alpha_i \left[y_i \left(\boldsymbol{w}^t \boldsymbol{x} + b\right) - 1\right]. \tag{3}$$

where $\alpha \geq 0$ denotes Lagrange-multiplier. Partial differentiations of Equation (3) by $\boldsymbol{w}$ and b are substituted for Equation (3), we obtain Equation (4).

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^t \boldsymbol{x}_j, \tag{4}$$

in subject to

$$\forall_i,\ \alpha_j \geq 0,\ \sum_{i=1}^{n} \alpha_i y_i = 0. \tag{5}$$

Here we denote α_i to maximize Equation (4) as α_i^*. The sample $\boldsymbol{x}_i$ with $\alpha_i^* > 0$ is called as Support Vector (SV), it exists on H_1 or H_2. The optimum of $\boldsymbol{w}$ denoted as $\boldsymbol{w}^*$ is obtained from the partial differentiations of Equation (3) and α_i^* by Equation (6).

$$\boldsymbol{w}^* = \sum_{i=1}^{n} \alpha^* y_i \boldsymbol{x}_i. \tag{6}$$

The optimum of b denoted as b^* is obtained from the Equation (7) with any $\boldsymbol{x}_s$ $(s \in SV)$.

$$b^* = y_s - \boldsymbol{w}^{*t} \boldsymbol{x}_s. \tag{7}$$

Finally, we obtain the discriminant function of SVM for linearly separable problem as Equation (8).

$$f(\boldsymbol{x}) = sign\left(\sum_{i \in SV} \alpha_i^* y_i \boldsymbol{x}_i^t \boldsymbol{x} + b^*\right). \tag{8}$$

We use kernel trick for non-linear problems in usual. The kernel function was radial basis function (RBF) in this study. We used the application program SVM^{light} to calculate the support vector machine [7]. The parameter C which is the trade-off between training error and margin is 1, and γ which is parameter in RBF kernel is 1.

3.2 Dataset

We used data of haemophilia A Mutation Database. The number of data in A1, A2, B, A3, C1 and C2 domain are 111, 131, 18, 107, 39 and 33 respectively. We used all data in each domain for training data and test data of SVM. We considered serious illness with less than 1% of factor VIII activity, and slight illness with more than 1% of one. We predicted the serious or slight illness of haemophilia A by SVM based on these data.

We used a distance between amino acids for each four amino acid physical-chemical parameters (Molecular volume, Hydropathy, Polar requirement and Isoelectric poin). The k-th distance between amino acid A_i and A_j is defined as

$$D_{ij}^{(k)} = |\, f_k(\mathrm{A}_i) - f_k(\mathrm{A}_j)\,|, \quad (k = 1, 2, 3, 4). \tag{9}$$

In this study, A_i is a normal amino acid, and A_j is a substituting amino acid.

Moreover, we used the calculated accessible area and fractional sidechain solvent accessibility in addition to physical-chemical parameters. The accessible area of the side chain was given in Angstroms squared. The fractional sidechain solvent accessibility is accessible to a 1.4 Angstrom squared probe, rolled over the model. A value of 0 = inaccessible, 1 = totally accessible.

4 Results

4.1 Prediction of the Severity

At first, we studied whether we could predict the severity of haemophilia A by using SVM. Furthermore, we compared the results of using all domain data and each domain data.

In Fig. 1, the horizontal axis is the false positive ratio, and the vertical axis is the true positive ratio. False positive means that the predicted result is positive (serious), but observed result is negative (slight). True positive means that both the prediction result and observed result are positive. False negative and true negative are similar to these. We calculated the ratios in cut-off point into every

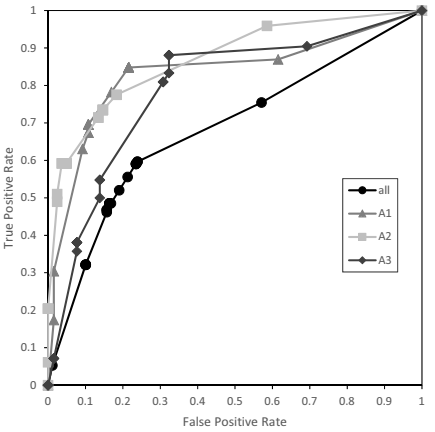


Fig. 1. The relationship between the false positive ratio and the true positive ratio in A1, A2 and A3 domain

0.1 from 0 to 1 using these values. We plot the relationship of the false positive ratio and the true positive ratio in Fig. 1. Such a figure is called ROC curve. ROC curve is used for a comparison of the inspection performance, which in the upper left indicates more superior performance.

In Fig. 1, circle mark shows the result of prediction using all domain data. The triangle mark shows the result of prediction using A1 domain data. The square mark shows the result of prediction using A2 domain data. The diamond mark shows the result of prediction using A3 domain data. Three curves of the using A1, A2 and A3 domain data are situated in upper left of the curve of the using all domain data. We ware not able to get a clear result in other domains. Therefore, this result suggested that the predictions by using A domain data are more superior performance than using all data.

4.2 Influence of Structural Parameter

Secondly, we studied influence of structural parameter to prediction of severity of haemophilia A. We predicted the severity using SVM with structural parameters which are accessible area and fractional accessibility.

In Fig. 2, 3 and 4 the horizontal axis is the false positive ratio, and the vertical axis is the true positive ratio. This figure shows result of prediction of serious or slight illness in A domain. The circle mark shows the result of prediction using A domain data without structural parameters. The triangle mark shows the result of prediction using A domain data with the accessible area parameter. The square mark shows the result of prediction using A domain data with the fractional accessibility parameter. Fig. 2, Fig. 3 and Fig. 4 are the result in A1, A2 and A3 domains respectively.

The curves of result using structural parameters laid on more upper left in every figures. Especially, the curve of using accessible area parameter is situated

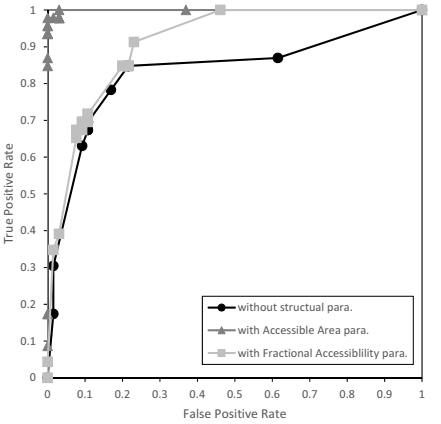


Fig. 2. The comparison between prediction with structural parameter and without structural parameter in A1 domain

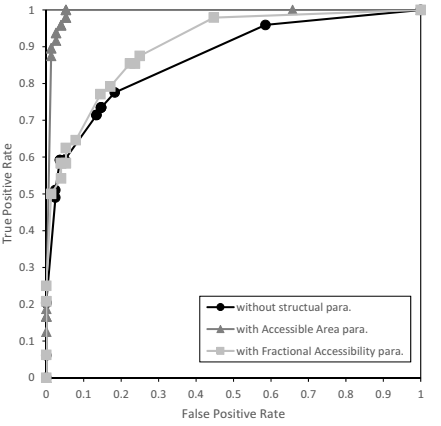


Fig. 3. The comparison between prediction with structural parameter and without structural parameter in A2 domain

on more upper left. Therefore, the results using structural parameters is more superior than the result without structural parameters. This result suggests that structural feature is the important to severity of haemophilia A.

4.3 Relationship between Domains

Finally, we studied relationships between domain in prediction of severity of haemophilia A. We predicted the severity by using SVM with some domain data for training and another domain data for test.

In Fig. 5 and Fig. 6, the horizontal axis is the false positive ratio, and the vertical axis is the true positive ratio. In Fig. 5, circle mark shows the result of prediction using A1 domain data only. The triangle mark shows the result

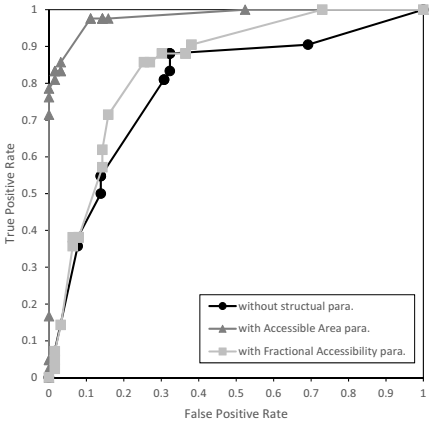


Fig. 4. The comparison between prediction with structural parameter and without structural parameter in A3 domain

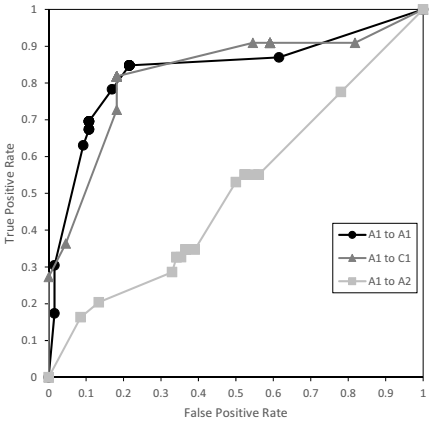


Fig. 5. The relationship between A1 domain and other domain

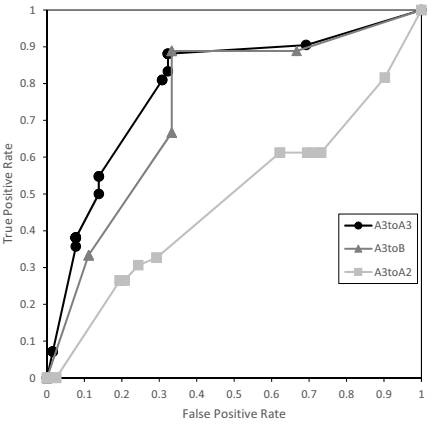


Fig. 6. The relationship between A3 domain and other domain

of prediction using A1 domain data for training and C1 domain data for test. These two curves are similar. Therefore, we suppose that the model of SVM in A1 domain can be applied to the prediction of severity in C1 domain. On the other hand, the square mark shows the result of prediction using A1 domain data for training and A2 domain data for test. This curve is not similar with other two curves.

In Fig. 6, circle mark shows the result of prediction using A3 domain data only. The triangle mark shows the result of prediction using A3 domain data for training and B domain data for test. These two curves are similar. Therefore, we suppose that the model of SVM in A3 domain can be applied to the prediction of severity in B domain. On the other hand, the square mark shows the result

of prediction using A3 domain data for training and A2 domain data for test. This curve is not similar with other two curves.

We could predict the severity of haemophilia B using others domain data which have homological structure in past study[4]. We know that there is approximately 30% homology of amino acid alignment between A1, A2 and A3 domain. However, we could not observe the good prediction of severity in A domains in this study.

5 Conclusion

We predicted the severity of haemophilia A by using SVM analysis in each A domain. In addition, we examined the influence of the structural parameter to prediction of severity in haemophilia A. In A1, A2 and A3 domain, we got the result that structural parameters influence the prediction. We analyzed the relationship between domains by SVM. As a result, A1 and C1, A3 and B have the relationship in the prediction of serious or slight illness in haemophilia A.

References

1. Bolton-Maggs, P.H.B., Pasi, K.J.: Haemophilias A and B. The Lancet 361, 1801–1809 (2003)
2. Furie, B., Furie, B.C.: The Molecular Basis of Blood Coagulation. Cell 53, 505–518 (1988)
3. Utsunomiya, M., Sakamoto, M., Furutani, H.: Regression Analysis of Amino Acid Substitutions and Factor IX Activity in Hemophilia B. Artificial Life and Robotics 13, 531–534 (2008)
4. Aoki, K., Yamamori, K., Sakamoto, M., Furutani, H.: Analysis of Genetic Disease Hemophilia B by Using Support Vector Machine. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part IV. LNCS, vol. 7666, pp. 476–483. Springer, Heidelberg (2012)
5. Gitschier, J., Wood, W.I., Goralka, T.M., et al.: Characterization of the Human Factor VIII Gene. Nature 312, 326–330 (1984)
6. Kemball-Cook, G.: The Haemophilia A Mutation, Structure, Test and Resource Site (HAMSTeRS), `http://hadb.org.uk/` (February 11, 2013) (updated)
7. Jachims, T.: SVM$^{\text{light}}$ Version 6.02 (2008), `http://svmlight.joachims.org/`
8. Schölkopf, B., Burges, C.J.C., Smola, A.J.: Advances in Kernel Methods. The MIT Press, London (1999)
9. Schölkopf, B., Tsuda, K., Vert, J.P.: Kernel Methods in Computational Biology. A Bradford Book. The MIT Press, London (2004)

An Innovative Fingerprint Feature Representation Method to Facilitate Authentication Using Neural Networks

Mark Abernethy and Shri M. Rai

Murdoch University, Perth Australia
{mark.abernethy,s.rai}@murdoch.edu.au

Abstract. Authentication systems enable the verification of claimed identity; on computer systems these are typically password-based. However, such systems are vulnerable to numerous attack vectors and are responsible for a large number of security breaches. Biometrics is now commonly investigated as an alternative to password-based systems. There are numerous biometric characteristics that can be used for authentication purposes, each with different levels of accuracy and positive and negative implementation factors. The objective of the current study was to investigate fingerprint recognition utilizing Artificial Neural Networks (ANNs) as a classifier. An innovative representation method for fingerprint features was developed to facilitate verification by ANNs. For each participant, the method required the alignment of their fingerprint samples (based on extracted local features), and the selection of 8 of these aligned features common to their samples. The six attributes belonging to each of the selected features were used for ANN input. Unlike the common usage, each participant had one dedicated ANN trained to recognize only their fingerprint samples. Experimental results returned a false acceptance rate (FAR) of 0.0 and a false rejection rate (FRR) of 0.0022, which were comparable to (and in some cases, slightly better than) other research efforts in the field.

Keywords: Authentication, Verification, Biometrics, Fingerprint Recognition, Artificial Neural Networks, Feature Representation, Pattern Recognition.

1 Introduction

Traditional authentication procedures require legitimate personal to supply a unique identifier (e.g. username) and a verification token (e.g. password). However, user defined passwords are low in entropy (i.e. randomness) and therefore easily 'cracked' [1]. Higher entropy passwords are more difficult to remember, and are commonly written down and left in the vicinity of the computer desk; this makes them vulnerable to loss or theft. Therefore contrary to popular belief, knowledge or possession of a verification token does not truly verify identity; it merely verifies the token holder, who may or may not be the legitimate identity.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 689–696, 2013.

Biometrics is an area of research now commonly investigated as an alternative to token-based authentication. Biometrics concerns the use of the physical traits and behavioural characteristics[1] that make each individual unique, and encompasses any personal characteristic that can be used to uniquely verify a person's identity [2].

A biometric authentication system essentially operates as a pattern recognition system [2]. Automated methods used in biometric authentication involve: the capture of biological data; the extraction of uniquely identifiable features from these data; the processing of extracted features into a format that can be stored and later retrieved—referred to as the 'registered template'; the comparison of a 'query template'[2] with the registered template. This comparison, or verification of fingerprints, is usually reliant on minutiae-based techniques. Artificial Neural Networks (ANNs) have also featured in fingerprint verification research (Sect 3). The typical usage of ANNs in this context is to train an ANN to differentiate between all subjects in a subject group. However, it becomes inconvenient and cumbersome to re-train an ANN every time a subject needs to be added to the subject group [3]. To resolve this issue, the current study adopted a different approach; to train one ANN for each subject.

Section 2 discusses fingerprint recognition and Sect 3 provides a brief review of related research. The research method is presented in Sect 4, and the results and conclusion are presented in Sects 5 and 6.

2 Fingerprint Recognition

The epidermal layer of the distal phalanx of a finger or thumb is covered with concentric raised friction ridges. Ridges (and the consequent furrows) eventuate in the most prominent characteristics of a fingerprint impression. A fingerprint is produced when the bulbous region of the distal phalanx makes contact with another surface, thus creating a duplicate impression of the existent characteristics of that finger tip [4]. Fingerprint characteristics are either global or local.

Global features are those fingerprint characteristics that are visible to the naked eye, and are described as follows [5]: *a.* Pattern area: the region of the fingerprint where ridge lines form clearly apparent and highly distinguishable shapes or patterns; *b.* Basic ridge pattern: the discernible patterns (located within the pattern area) made by ridge lines that have been defined into categories or classes; *c.* Core point (if existent): the upper most point (in relation to the tip of the finger) of the inner most ridge line; *d.* Delta point (if existent): when a single ridge abruptly bifurcates into two and the diverging ridges depart in opposite directions; *e.* Ridge count: the number of ridges crossing the imaginary line segment between a core point and a delta point; *f.* Minutiae count: the total number of minutia.

[1] Eg: fingerprints, retinal/iris patterns, voice/speech patterns, facial image patterns.

[2] The query template is formulated from a sample obtained during attempted authentication, and processed in exactly the same manner as the registered template.

Global features are commonly used to categorize fingerprints into general classes [6]. These are broadly classified as: arch, loop, and whorl. Importantly, global features are insufficiently distinctive enough for verification purposes.

Local features are not visible to the naked eye. Fingerprint ridges are not continuous straight lines; they may break, fork, change direction, or terminate [5]. The point of discontinuity is called a minutia point (plural term minutiae). There are five attributes of a minutia: *a.* Primary Type[3]: ridge termination, when a ridge ends abruptly; ridge bifurcation, when a ridge divides into two or more individual ridges; *b.* Position: the location of the minutia, determined as (x, y) coordinates in a two dimensional coordinate system; *c.* Spatial frequency: the average distance between ridges in the neighbourhood of a minutia; *d.* Orientation: the angle between the tangent to the ridge at a minutia position and the horizontal axis (at right angles to the vertical axis of the finger); *e.* Curvature: the rate of change of the ridge orientation as the ridge approaches a minutia.

Typically, all fingers have a different number of minutiae. Even if two fingers have the same number of minutiae, they will be in different relative positions. The relative position of minutiae forms a unique configuration or pattern. This pattern and the other minutiae attributes are used for verification [6].

Previous studies have shown that successive scans of the same finger produce images that are rarely identical [6]. Reasons for variability can be sensor inaccuracy (resulting in missing data or introduced artifacts), instrument noise, adverse ambient conditions, changes in physiological characteristics, and the elasticity of the epidermal layer of the finger [6]. However, the distinctive feature patterns will still be evident (though typically subject to local distortions).

3 Related Research

There are two common steps required in feature matching techniques: feature alignment (registration) and the matching process. Registration attempts to align a query feature set with those of the registered feature set.

A number of the research efforts utilized a curved line segment in both feature sets as a basis for the alignment process—with both line segments originating at a respective reference minutia [7,10]. Typically, nominated points along the ridge were determined and utilized in the alignment process. This process was performed for a localized area of the fingerprints under examination.

Jiang and Yau (2000), utilized the configuration of local and global fingerprint structures to first align two samples; the same structure information was then used in the matching process [8]. Lee et al. (2002), calculated the normalized ridge distance of a fingerprint and used this in conjunction with the configuration of structures in a localized area (within a prescribed circle of arbitrary radius) [9]. Tong et al. (2005), based alignment on points along ridges (in the local area) adjacent to the ridge originating at a reference minutia, to construct adjacent feature vectors [11].

[3] Variations of the two primary types do occur, and include: independent ridge, dot or island, and enclosure.

Qi and Wang (2005), based alignment on a sampling of points along imaginary lines whose orientations were at nominated angles around a reference minutia [12]. The angles around the minutia were specified as $\theta_1, \theta_1 + (2\pi/3), \theta_1 + (4\pi/3)$, where θ_1 was the orientation of the ridge originating at the reference minutia (again a local area alignment). Jie et al. (2006), utilized core points (refer Sect 2) in the alignment process [13].

The matching process conducted by a number of the above research efforts utilized converted polar coordinates. According to Jain et al. (1997), polar coordinates minimize the radial distortion properties associated with local area non-linear deformations [7]. Also, reduced error in the calculation of the rotational factor (during alignment) is more likely if polar coordinates are used.

Bounding boxes are commonly utilized during the determination of matching points. A bounding box allows two points to be considered to match even if their aligned locations are not precisely the same; this allows a small degree of flexibility to compensate for non-linear distortions.

Examples of research investigating ANNs for verification are Kumar and Deva Vikram (2010) and Sadi and Tanij (2011) [14,15]. Kumar and Deva Vikram (2010) enhanced a fingerprint image to isolate minutiae, and reduced it to a 15 x 15 matrix. In each matrix the intensity values of each pixel were summed; this sum was then normalized to the interval [0,1] and applied to a Multi-Dimensional ANN. Whilst novel, the approach may not preserve the uniqueness provided by the local feature configuration [14]. Sadi and Tanij (2011) used an edge detection technique to obtain binarized images, which were applied to ANNs (back propagation network) for verification (achieved by comparing the weight matrices of trained networks) [15].

ANNs have also been used for classification purposes [16,17]. However, classification performance measures a recognition rate, which cannot be compared to those used to measure verification performance—the False Acceptance Rate (FAR) and False Rejection Rate (FRR) (refer Sect 5).

Table 1 demonstrates the reported FAR and FRR for the experiments reviewed in this section.

Table 1. Summary results for reviewed papers in verification research

Reviewed Paper	FAR	FRR
Jiang and Yau, 2000	0.0	0.0997
Lee et al., 2002	0.0002	0.1666
He et al., 2003	0.0001	0.045
Tong et al., 2005	0.00001	0.07
Qi and Wang, 2005	0.0325	0.0605
Jie et al., 2006	0.00001	0.001
Kumar and Deva Vikram, 2010	0.0113	0.015
Sadi and Tanij 2011	0.0	0.172

4 Research Method

From the review in Sect 3, it is apparent that fingerprint recognition typically involves two commensurate and integrated processes: an alignment and a matching process. The approach adopted in the current experiment differs in that alignment was carried out at the feature level as a separate process. A feature selection process was then applied to the aligned local fingerprint features. This selection process determined the input features for the ANN, and the ANN was then trained to recognize only one person and reject all others.

Ninety participants were recruited from the authors' institution. Fingerprint scans (of participants right index finger) were captured using the Digital Persona U.ARE.U 4000 optical fingerprint scanner; participants provided 140 scans each. The Verifinger SDK was used to extract global and local feature information from each scan, which were subsequently used in the experiment. These extracted data were written to a uniquely named file for each participant.

The newly developed fingerprint feature representation method (described below) required fingerprint feature alignment (registration) and matching of corresponding minutiae between feature sets. The registration process adopted the point pattern matching algorithm proposed by Van Wamelin et at. [18]. They tested the accuracy of the algorithm, and showed that it was rotation, scale and translation invariant and robust to missing data and/or introduced artifacts. The matching process was influenced by the intention to use dedicated ANNs to perform verification of each participants fingerprint feature data. To facilitate ease of ANN training and testing, all input vectors to the ANNs were required to be of a standard length, which was achieved via a feature selection process.

For fingerprint data, accurately registered features should not exhibit noise or variability (to any great degree). However as discussed in Sect 2, they do exhibit variability of a different nature (specifically variation in the number of features for the same person's scans from sample to sample, and the number of features from person to person). Therefore, formulating ANN input vectors of a standard length required a feature selection process.

Typically, the x and y coordinates and the orientation are the attributes used in minutiae matching algorithms [6]; with the x and y coordinates considered as 2 separate attributes, this results in 3 attributes. For the current experiment it was considered that the use of all available local feature attributes (refer Sect 2) might be more distinctive, and more beneficial to the pattern recognition task. Thus there were 6 available attributes for each local feature. As far as is known, no other study has incorporated all 6 local feature attributes.

Historically, fingerprint matching for law enforcement requires a 12 minutia point match to be considered incontrovertible. However as all 6 attributes were being utilized, a quantity of 8 local features (minutiae) was considered adequate for this experiment. Therefore, the selection process involved the determination of 8 local features occurring in all 140 samples for a participant; resulting in a total of 48 local feature attributes (for each sample for each participant).

The selected features were extracted and normalized according to the minimax method, and written to uniquely named metrics files for each participant.

Fifty participants metrics files were randomly assigned to the training group, and the remaining 40 to the non-training group.

An ANN was trained for each training group member (refer Sect 1) and an input file for training was generated as follows: 30 samples were randomly chosen (and removed) from that member's metrics file, for the positive training case; 1 sample was randomly chosen (but not removed) from each of the other training group members metrics files (1 x 49), for the negative training case.

A cross validation file (to assist training) was generated for each training group member. Ten samples were randomly chosen (and removed) from the same file from which the 30 training samples were chosen. Therefore, the 50 training group members training input files consisted of 79 samples each, and their validation files consisted of 10 samples each[4].

The objective of the training phase was to obtain a registered template (for each training group member) associated with their training input file. A two layer Multi-Layer Perceptron architecture (with back propagation) was used as a pattern classifier; thus a dedicated ANN was trained for each training group member, applying their training input file. Once the ANNs were trained, the weights from each member's trained ANN were saved and used as their registered template; this meant a specific ANN configuration for that member.

A testing input file was generated for each training group member as follows: 100 samples (those not used in training/validation) were used to test for Type II errors (incorrect rejection); 100 unused samples from each of the other training group members (100 x 49), and 140 samples from each of the non-training group members (140 x 40), were used to test for Type I errors (incorrect acceptance).

Therefore, the 50 training group members testing input files consisted of 10,600 samples each. For the testing phase, weights representing a member's registered template were loaded into an ANN, thus utilizing the specific ANN configuration for that member. The testing input file for that member was applied to the ANN, and the predicted outcomes were used to calculate results.

5 Results

The classification outcome for an authentication system involving biometrics is the likelihood that two samples belong to the same individual [6]. This necessitates a subjective decision based on whether or not the predicted outcome (in this case, ANN output) should be accepted or rejected. This determination may be termed 'the final classification decision', and typically involves the use of a 'decision threshold' applied to the predicted outcome.

The two performance variables used to measure experimental results were the False Acceptance Rate (FAR)—the number of Type I errors divided by the number of samples available to test for Type I errors—and the False Rejection Rate (FRR)—the number of Type II errors divided by the number of samples available to test for Type II errors.

[4] With 40 samples removed from members metrics files and used for training purposes, 100 samples remained per training group member to test Type II errors.

The individual results demonstrated that all training group members achieved a FAR of 0.0. This meant that all training group members had no impostor samples (out of 10,500 each) accepted as their own. It was also demonstrated that only three training group members registered a non-zero FRR. The highest FRR was 0.05, which meant that for two members 5 samples (of 100 genuine samples) were incorrectly rejected; the other non-zero FRR was 0.01, which meant that only 1 genuine sample was incorrectly rejected.

The average FAR was 0.0 and the average FRR was 0.0022. This means that for all training group members, no impostor samples were incorrectly accepted, and there was approximately 2 in 1,000 genuine samples incorrectly rejected.

In comparison to the results shown in Table 1 Sect 3, the current experiment returned FAR results comparable to Jiang and Yau (2000) and Sadi and Tanij (2011), and better than the other reviewed works. Jiang and Yau (2000) used 1,503 samples (per participant) to test Type I errors, Sadi and Tanij (2011) used 90, whereas the current experiment used 10,500. The FRR results were comparable to results achieved by Jie et al. (2006), and better than the other reviewed works. Jie et al. (2006) used 10 samples (per participant) to test Type II errors, whereas the current experiment used 100.

Of the 10,500 samples tested per training group member for Type I errors, 5,600 comprised 140 samples from each of the 40 non-training group members. Samples from these participants data were not seen at all by the ANNs during the training process, so the results should be considered generalizable.

An inherent property of performance measurement in biometric systems is the trade-off between the FAR and FRR; they are functions of the decision threshold [6]. If the threshold is decreased—to provide tolerance—the FAR increases. Conversely, if it is increased—to increase security—the FRR increases. Thus if ANN over training was evident, FAR scores of 0.0 would result in higher FRR scores and vice versa. As this did not occur in the current experiment, it is believed that the excellent results were attributable to correct classification rather than ANN over training.

6 Conclusion

The discussion presented in Sect 5 demonstrated that the newly developed fingerprint feature representation method achieved results comparable to (and in some cases, slightly better than) most previous studies involving fingerprint recognition. The experimental methodology involved more participants than many studies reviewed in Sect 3, and there were generally many more samples provided by participants in the current study.

The innovative fingerprint feature representation method developed for this experiment, and the subsequent feature selection process, performed extremely well. Using a dedicated ANN per person, simplified ANN training as issues relating to inter-class versus intra-class variances were minimized. Though the treatment of data for this experiment was heavily influenced by the utilization of

ANNs as a pattern classifier, it could be expected that there may be other appropriate methods for fingerprint feature representation that could return equally good results.

References

1. Schneier, B.: Secrets & Lies: Digital Security in a Networked World. John Wiley and Sons, Inc. (2000)
2. Jain, A., Ross, A., Prabhakar, S.: An Introduction to Biometric Recognition. IEEE Transactions on Circuits and Systems for Video Technology 14(1), 4–20 (2004)
3. Yager, N., Amin, A.: Fingerprint Verification Based on Minutiae Features: A Review. Pattern Analysis Applications 7(1), 94–113 (2004)
4. Galton, F.: Finger Prints. Macmillan and Company (1892)
5. Abernethy, M.: User Authentication Incorporating Feature Level Data Fusion of Multiple Biometric Characteristics. Murdoch University, Australia (2011), `http://researchrepository.murdoch.edu.au/10609/`
6. Maltoni, D., Maio, D., Jain, A., Prabhakar, S.: Handbook of Fingerprint Recognition. Springer (2003)
7. Jain, A., Hong, L., Bolle, R.: On-Line Fingerprint Verification. IEEE Transactions on Pattern Recognition and Machine Learning 19(4), 302–314 (1997)
8. Jiang, X., Yau, W.: Fingerprint Minutiae Matching Based on the Local and Global Structures. In: Proceedings of the 15th International Conference on Pattern Recognition, vol. 2, pp. 1038–1041 (2000)
9. Lee, D., Choi, K., Kim, J.: A Robust Fingerprint Matching Algorithm Using Local Alignment. In: Proceedings of the 16th International Conference on Pattern Recognition, vol. 3, pp. 803–806 (2002)
10. He, Y., Tian, J., Luo, X., Zhang, T.: Image Enhancement and Minutiae Matching in Fingerprint Verification. Pattern Recognition Letters 24, 1349–1360 (2003)
11. Tong, X., Huang, J., Tang, X., Shi, D.: Fingerprint Minutiae Matching Using The Adjacent Feature Vector. Pattern Recognition Letters 26, 1337–1345 (2005)
12. Qi, J., Wang, Y.: A Robust Fingerprint Matching Method. Pattern Recognition 38, 1665–1671 (2005)
13. Jie, Y., Yi Fang, Y., Renjie, Z., Qifa, S.: Fingerprint Minutiae Matching Algorithm for Real Time System. Pattern Recognition 39(1), 143–146 (2006)
14. Kumar, R., Deva Vikram, B.: Fingerprint Matching Using Multi-Dimensional ANN. Engineering Applications of Artificial Intelligence 23, 222–228 (2010)
15. Sadi, M., Kanij, T.: Fingerprint Verification: A Comparison of Three Approaches. In: Defense Science Research Conference and Expo IEEE, pp. 1–5 (2011)
16. Arantes, M., Ide, A., Saito, J.: A System for Fingerprint Minutiae Classification and Recognition. In: Proceedings 9th International Conference on Neural Information Processing, vol. 5, pp. 2474–2478 (2002)
17. Xiangrong, L., Guohui, W., Xiangjiang, L.: Neural Network Based Automatic Fingerprints Classification Algorithm. In: International Conference of Information Science and Management Engineering, vol. 1, pp. 94–96 (2010)
18. Van Wamelen, P., Li, Z., Iyengar, S.: A Fast Expected Time Algorithm for The 2-D Point Pattern Matching Problem. Pattern Recognition 37, 1699–1711 (2004)

Application of the Dynamic Binary Neural Network to Switching Circuits

Yuta Nakayama, Ryota Kouzuki, and Toshimichi Saito*

Hosei Univ., Tokyo, 184-8584 Japan
ryota.kouzuki.4a@stu.hosei.ac.jp

Abstract. This paper studies application of the dynamic binary neural network to control signal of switching circuits. The network is characterized by the signum activation function and ternary weighting parameters. In the application, the teacher signal is one binary periodic orbit corresponding to the controls. The learning algorithm is based on the genetic algorithm. As an application object, we consider a control signal of the basic matrix converter. Performing a basic numerical experiment, we have confirmed that the teacher signal is stored successfully and is stabilized automatically.

Keywords: supervised learning, binary neural networks, power electronics.

1 Introduction

The dynamic binary neural network (DBNN) is a digital dynamical system constructed by applying the delayed feedback to the three-layer neural network [1] [2]. The DBNN has signum activation function and can exhibit various binary periodic orbits (BPOs). The DBNN is suitable for precise numerical analysis of the dynamics and hardware implementation by digital circuits.

This paper studies an application of the DBNN to control signal of switching circuits. In the application, the teacher signal is one BPO that corresponds to the control signal. The learning algorithm is based on the genetic algorithm (GA). In the algorithm, chromosomes correspond to the ternary weighting parameters. If the BPO is stored successfully, we measure a basic feature quantity that characterizes domain of attraction (DOA [3]) to the teacher signal BPO: it is basic to consider the stability of the BPO. Although there exist many examples of the switching circuit, we consider the basic matrix converter that converts three-phase ac input signals to the other three-phase ac output signals having different frequency. The matrix converter is an important circuit in the power electronics [4] [5] and has switches whose control signal corresponds to the teacher signal BPO. Performing numerical experiment, we have confirmed that the GA-based learning can store the BPO into a DBNN with simple structure. We have also confirmed that the stored BPO can be stabilized automatically. These results can

* This work is supported in part by JSPS KAKENHI#24500284.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 697–704, 2013.

be basic information not only for application to various switching circuits but also for analysis/synthesis of various digital dynamical systems. Note that the DBNN is a digital dynamical system with various real/potential applications as digital dynamical systems [6]-[10]. For novelty of this paper, we note that Refs. [11] [12] have discussed neither application to the matrix converters nor general discussion on application to switching circuits.

2 Dynamic Binary Neural Networks

The DBNN is constructed by applying the delayed feedback to the three-layer network as shown in Fig. 1. Each hidden neuron has signum activation function and the DBNN dynamics is described by the following:

$$\begin{aligned} x_i(t+1) &= \mathrm{sgn}\left(\sum_{j=1}^{M} w_{ij}^o \xi_j(t) - T_i^o\right),\ i = 1 \sim N \\ \xi_j(t) &= \mathrm{sgn}\left(\sum_{i=1}^{N} w_{ji} x_i(t) - T_j\right),\ j = 1 \sim M \\ \mathrm{sgn}(x) &= \begin{cases} 1 & \text{for } x \geq 0 \\ -1 & \text{for } x < 0 \end{cases} \end{aligned} \tag{1}$$

where $\boldsymbol{x}(t) = (x_1(t), \cdots, x_N(t))$, $x_i(t) \in \{-1, 1\} \equiv \boldsymbol{B}$, is an N-dimensional binary state vector at a discrete time t. $\boldsymbol{\xi}(t) \equiv (\xi_1(t), \cdots, \xi_M(t))$, $\xi_j(t) \in \boldsymbol{B}$, is an M-dimensional hidden output vector at t. The hidden neurons are characterized by the ternary connection parameters w_{ij} and integer threshold parameters T_j:

$$w_{ji} \in \{-1, 0, 1\},\ T_j \in \{-N-1, -N, \cdots, N, N+1\} \tag{2}$$

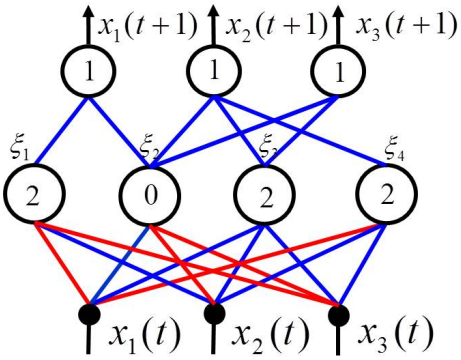


Fig. 1. Dynamic Binary Neural Network (DBNN) for $N = 3$. Blue and red segments represent connection value 1 and -1, respectively. $w_{ji} = 0$ means no connection.

The output neurons are characterized by the binary connection parameters w_{ij}^o

$$w_{ij}^o \in \{0,1\},\ T_i^o = 1 - \sum_{j=1}^{M} |w_{ji}^o| \tag{3}$$

Note that the integer thresholds T_j^o are determined uniquely by w_{ij}^o. In order to clarify the meaning of this output neuron, we define a subset of hidden outputs that connect to the i-th output neuron: $H_i \equiv \{\xi_j \mid w_{ji}^o = 1\}$. In the DBNN in Fig. 1, $H_1 = \{\xi_1, \xi_2\}$, $H_2 = \{\xi_2, \xi_3, \xi_4\}$ and $H_3 = \{\xi_2, \xi_3\}$. The output neuron operates equivalently to the logical OR of all the elements in H_i:

$$x_i(t+1) = \begin{cases} -1 \text{ if all the elements of } H_i \text{ are -1} \\ 1 \quad \text{otherwise} \end{cases} \tag{4}$$

We refer to H_i as the i-th hidden set. The DBNN dynamics is controlled by the three kinds of parameters w_{ji}, T_i and w_{ij}^o. The DBNN realizes a mapping of N-D binary vector. For simplicity, Eq. (1) is abbreviated by

$$\begin{aligned} &\boldsymbol{x}(t+1) = \boldsymbol{F}_D(\boldsymbol{x}(t)),\ \boldsymbol{F}_D : \boldsymbol{B}^N \to \boldsymbol{B}^N \\ &\boldsymbol{F}_D \equiv (F_{D1}, F_{D2}, \cdots, F_{DN}) \end{aligned} \tag{5}$$

As a present state $\boldsymbol{x}(t)$ is input, the DBNN outputs the next state $\boldsymbol{x}(t+1)$. Repeating in this manner, the DBNN can output a sequence of binary vectors. Since the domain of F_D consists of finite elements, the steady state of the DBNN is a binary periodic orbit (BPO) as illustrated in Fig. 1 caption. Here we give basic definitions of the BPO.

A vector $\boldsymbol{x} \in B^N$ is said to be a binary periodic vector (BPV) with period p if $\boldsymbol{F}_D^p(\boldsymbol{x}_p) = \boldsymbol{x}_p$ and $F_D^k(\boldsymbol{x}_p) \neq \boldsymbol{x}_p$ for $k < p$ where $\boldsymbol{F}_D^p$ is the p-fold composition of $\boldsymbol{F}_D$ ($\boldsymbol{F}_D^p = \boldsymbol{F}(\boldsymbol{F}_D^{p-1}(\boldsymbol{x}))$). A sequence of the BPVs, $\{\boldsymbol{F}_D(\boldsymbol{x}_p), \cdots, \boldsymbol{F}_D^p(\boldsymbol{x}_p)\}$, is said to be a binary periodic orbit (BPO) with period p.

In this paper, we consider basic learning problem: finding suitable parameter values of w_{ji}, T_j and w_{ij}^o in order to store one desired BPO with period T:

$$\boldsymbol{z}(t+T) = \boldsymbol{z}(t),\ t = 1, 2, 3, \cdots \tag{6}$$

We use the T binary vectors, $\boldsymbol{z}(1)$ to $\boldsymbol{z}(T)$, as the actual teacher signal. There exist many methods to implement such a BPO. However, this paper considers several basic problems: making the DBNN configuration as simple as possible and investigation of the domain of attraction (DOA) to the BPO. The DOA is important to clarify the stability of the BPO. We give basic definition for the DOA.

A vector $\boldsymbol{x}_e \in I_N$ is said to be an eventually periodic vector (EPV) if it is not a BPV and there exists some integer m such that $\boldsymbol{F}_D^m(\boldsymbol{x}_e)$ is a BPV. The orbit started from the EPP falls into either BPO. The EPPs construct the domain of attraction (DOA) to the BPO. As a basic measure to characterized the DOA to the stored BPO, we present the convergence rate to the BPO:

$$\text{CR} = \frac{\#\text{EPV to the BPO}}{2^N - \#\text{BPV of the BPO}}$$

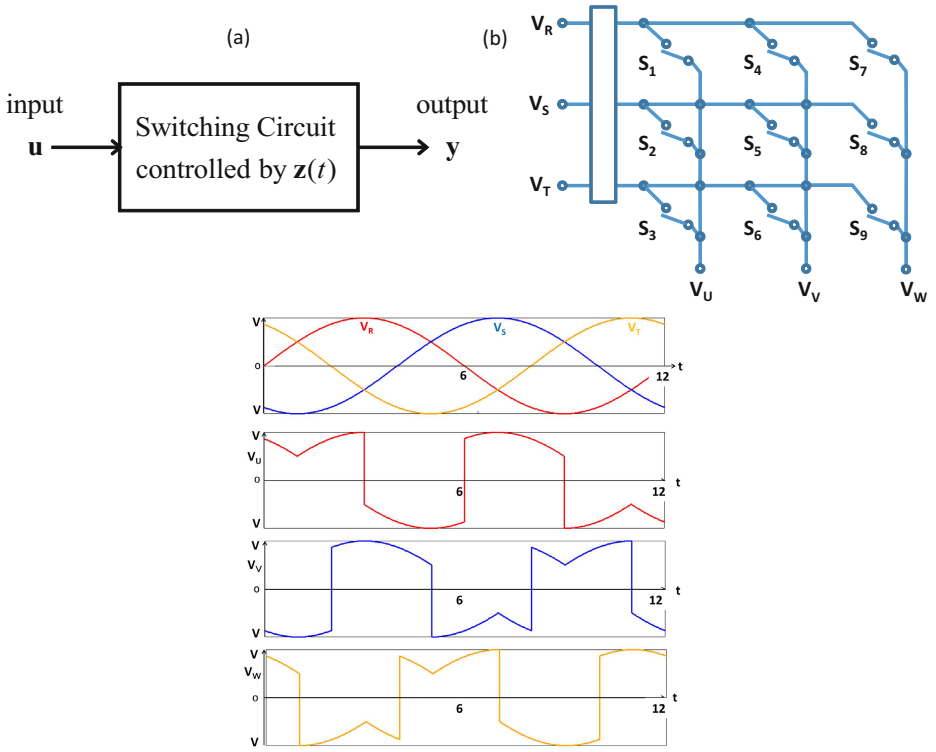


Fig. 2. Switching circuits controlled by a BPO. (a) switching circuit. (b) AC/AC matrix converter, The input three-phase voltages with period T (V_R, V_S, V_T). The output is three-phase voltages with period T (V_U, V_V, V_W).

If CR=1 then all the initial points fall into the BPO and the DOA is the largest. If CR=0 then no initial point fall into the BPO except for BPVs of the BPO and the DOA is the smallest. We can calculate the #EPV and #BPV directly if N is not too large. Note that the DOA of BPO has not been studied sufficiently in existing works.

3 Application to the Matrix Converter

Although there are many practical examples of the teacher signal BPOs, we consider the BPO in the switching circuits. Figure 2 (a) illustrates a circuit that converts an input u to an output y via a switching circuit consisting of N switches. The switch=on and =off can be symbolized by 1 and -1, respectively; and the operation of the switching are represented by a dynamic binary sequence. In the steady state, the switching operation is described by some BPO that can be a teacher signal of the DBNN. Examples of such switching circuits are many and the switching power converters are typical ones. As is well known, there

exists four kinds of power converters: dc-dc, dc-ac, ac-dc and ac-ac converters [4]. This paper considers a basic ac-ac converter as shown in Fig. 2 (b). This is a matrix converter that converts a three-phase ac input into some three-phase ac output via nine switches [5]. Table 1 shows the teacher signal BPO with period 12: $\boldsymbol{z}(t) = (z_1(t), \cdots .z_9(t))$ and $\boldsymbol{z}(t+12) = \boldsymbol{z}(t)$. This BPO controls the nine switches such that $z_i(t) = +1$ and $z_i(t) = -1$ correspond to $S_i(t)$ =on and $S_i(t)$ =off, respectively, where $i = 1 \sim 12$. Using this switching signal, the 3-phase ac inputs V_R, V_S, V_T with period 12 are converted into the 3-phase ac outputs V_U, V_V, V_W with period 6. Figure 2 shows the three-phase output waveforms. Filtering them, we obtain the 3-phase ac waveforms with period T.

Applying the GA-based learning in Appendix, we can store the BPO into the DBNN. Figure 3 shows an example of the DBNN with 15 hidden neurons. The parameter values and hidden sets are shown in Tables 2 and 3. In order to visualize the dynamics of the DBNN, we introduce the Gray-code-based return map (Gmap). First, we use the Gray code to express the domain of the DBNN $\boldsymbol{B}^N$. Let $G_N \equiv (G_0, \cdots, G_{2^N})$ be the Gray code of the $\boldsymbol{B}^N$. The G_N is equivalent to the set of rational numbers in $[0, 1)$ with denominator 2^N: $L_N \equiv (0/2^N, 1/2^N, \cdots, (2^N - 1)/2^N)$. We then notice that the dynamics of the DBNN can be simplified into the Gmap:

$$\theta(t+1) = F_G(\theta(t)),\ \theta(t) \in L_N. \tag{7}$$

Figure 4 shows the Gmap of the DBNN in Fig. 3. In the Gmap, we can see one red periodic orbit with period 12 that corresponds to the teacher signal BPO. In this example, we can confirm that the BPO is stored successfully and all the initial points fall into to BPO: CR=1. Note that the teacher signal BPO does not include information of the DOA, this BPO is stabilized automatically. If CR is large, the BPO has wide DOA and is suitable for robust circuit operation. Performing 100 trials of the learning with different initial condition of the GA, the algorithm can store the BPO in all the trials and we have obtained the following statistic data.

hidden neurons: max=17, min=14, avg=15.2
CR to the BPO: max=1, min=0.11, avg=0.45

4 Conclusions

Application of the DBNN to switching circuit has been studied in this paper. The teacher signal BPO corresponds to a control signal of switches in the circuits. As an example of the circuits, we consider the matrix converter. Applying the GA-based learning, we can store the BPO into the DBNN. Using the Gmap, the storage of BPO and its local stability (DOA) are visualized.

Future problems include analysis of learning process, mechanism of automatic stabilization, reduction of the number of hidden neurons application to various switching circuits and hardware implementation of the DBNN.

Appendix

Here we explain an outline of the GA-based algorithm that is an improved version of the algorithm in [1]. Our algorithm aims at finding parameters in order to store the teacher signal BPO. Note that the signum activation function prohibits the gradient descent method, the brute force is very hard to find the parameters w_{ij} in search space of 3^{NM} candidates and application of the logical synthesis often causes many redundant hidden neurons [1] [11]. These are reasons why we use the GA. Let an unknown Boolean function $z_i(t+1) = G_i(\boldsymbol{z}(t))$ govern the i-th element of the teacher signal BPO. Let $\boldsymbol{G} \equiv (G_1, \cdots, G_N)$. For the Boolean function G_i, $i = 1 \sim N$, an element $\boldsymbol{x} \in \boldsymbol{B}^N$ is said to be a true vertex if $G_i(\boldsymbol{x}) = 1$. Our learning algorithm tries to separate the true vertices of the teacher signal by the separating hyper planes: $\mathrm{SHP}_j = \sum_{i=1}^{N} w_{ji}x_i - T_j = 0$, $j = 1 \sim M$, where j is the index of the SHP and M is the number of SHPs (i.e., the number of hidden neurons) determined after the learning. The SHP_j corresponds to the j-th hidden neuron.

Step 1: (initialization) let $j = 1$.

Step 2: Find some component G_k of $\boldsymbol{G}$ that has the largest number of the true vertices.

Step 3: For G_k, we apply the GA having M_g chromosomes $\{\boldsymbol{C_1}, \cdots, \boldsymbol{C_{M_g}}\}$ each of which is a candidate of the $\{w_{k1}, \cdots, w_{kN}\}$. One of the initial chromosomes is selected from true vertices of teacher signals and other chromosomes are set randomly. It guarantees separation of at least one true vertex. The fitness is the number of separated true vertices when the SHP_j is applied to all the components of $\boldsymbol{G}$[1]. For each chromosome, the parameter T_j is determined to give the maximum fitness. Repeating the elite strategy and ranking selection, the chromosomes are evolved. The two-point crossover is applied with probability P_c and the one point mutation is applied with probability P_m. After G_{max} times evolution, the SHP_j is determined.

Step 4: For all the components G_l in which true vertices are separated by the SHP_j, the j-th hidden output ξ_j is added to the l-th hidden sets $\boldsymbol{H}_l$. All the separated true vertices are declared as "don't care".

Step 5: Let $j = j + 1$, go to Step 2 and repeat until all the true vertices are separated.

For the teacher signal BPO in Table 1, we have selected the following values after trial-and-errors: $M_g = 50$, $P_c = 0.8$, $P_m = 0.1$, and $G_{max} = 50$.

[1] The fitness in [1] is the number of the separated true vertices in each SHP. It causes redundant hidden neurons.

Table 1. Teacher signal BPO with period 12: $\boldsymbol{z}(t+12) = \boldsymbol{z}(t)$

$\boldsymbol{z}(1)$	$(-1,-1,+1,-1,+1,-1,-1,-1,+1)$
$\boldsymbol{z}(2)$	$(+1,-1,-1,-1,+1,-1,-1,+1,-1)$
$\boldsymbol{z}(3)$	$(+1,-1,-1,+1,-1,-1,-1,+1,-1)$
$\boldsymbol{z}(4)$	$(-1,-1,+1,+1,-1,-1,-1,-1,+1)$
$\boldsymbol{z}(5)$	$(-1,-1,+1,+1,-1,-1,+1,-1,-1)$
$\boldsymbol{z}(6)$	$(-1,-1,+1,-1,-1,+1,-1,+1,-1)$
$\boldsymbol{z}(7)$	$(-1,+1,-1,-1,-1,+1,-1,+1,-1)$
$\boldsymbol{z}(8)$	$(-1,+1,-1,+1,-1,-1,+1,-1,-1)$
$\boldsymbol{z}(9)$	$(-1,+1,-1,-1,+1,-1,+1,-1,-1)$
$\boldsymbol{z}(10)$	$(+1,-1,-1,-1,-1,+1,+1,-1,-1)$
$\boldsymbol{z}(11)$	$(+1,-1,-1,-1,-1,+1,+1,-1,-1)$
$\boldsymbol{z}(12)$	$(+1,-1,-1,-1,-1,+1,-1,-1,+1)$

Table 2. Parameters w_{ji} and T_j after the learning

j	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	w_{j6}	w_{j7}	w_{j8}	w_{j9}	T_j
1	0	+1	−1	−1	0	+1	−1	−1	+1	4
2	+1	−1	+1	−1	+1	−1	0	0	0	3
3	−1	+1	−1	+1	0	+1	+1	+1	+1	1
4	−1	−1	+1	−1	−1	0	+1	0	−1	4
5	−1	−1	−1	+1	0	0	−1	+1	+1	2
6	0	−1	−1	−1	−1	0	+1	0	−1	5
7	−1	+1	−1	0	+1	−1	0	+1	−1	4
8	0	0	−1	0	−1	−1	−1	+1	0	2
9	−1	−1	0	+1	0	−1	−1	0	−1	3
10	0	0	+1	−1	−1	+1	0	+1	+1	1
11	0	+1	+1	−1	0	−1	−1	+1	+1	2
12	0	+1	0	0	0	0	−1	−1	+1	3
13	0	−1	0	+1	0	−1	−1	−1	+1	5
14	−1	+1	0	+1	−1	0	+1	−1	−1	6
15	+1	−1	−1	0	−1	−1	+1	0	0	3

Table 3. Hidden Sets after the learning

H_1	$\{\xi_2, \xi_6, \xi_7\}$
H_2	$\{\xi_3, \xi_{10}\}$
H_3	$\{\xi_5, \xi_9, \xi_{12}\}$
H_4	$\{\xi_5, \xi_8\}$
H_5	$\{\xi_1, \xi_{11}, \xi_{12}, \xi_{14}\}$
H_6	$\{\xi_4, \xi_6, \xi_7\}$
H_7	$\{\xi_3, \xi_7, \xi_{13}\}$
H_8	$\{\xi_2, \xi_4\}$
H_9	$\{\xi_1, \xi_6, \xi_{15}\}$

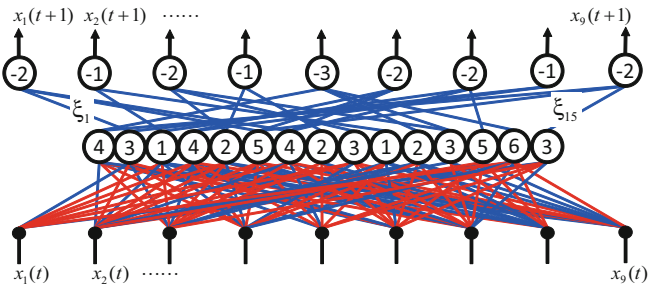


Fig. 3. Configuration of DBNN after the learning. Blue and red segments represent connection value 1 and -1, respectively.

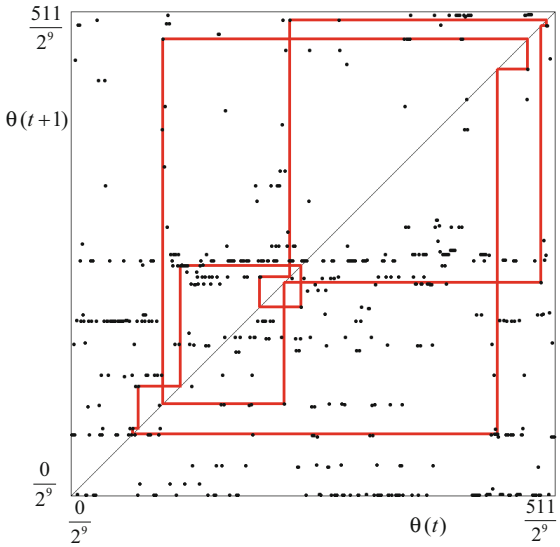


Fig. 4. Gmap of DBNN in Fig. 3 with BPO (red orbit) with period 12

References

1. Ito, R., Nakayama, Y., Saito, T.: Analysis and Learning of Periodic Orbits in Dynamic Binary Neural Networks. In: Proc. IEEE-INNS Int'l Joint Conf. Neural Networks, pp. 502–508 (2012)
2. Kouzuki, R., Suzuki, T., Saito, T.: Learning of Periodic Attractors in Simple Dynamic Binary Neural Networks. In: Proc. NDES, pp. 177–180 (2012)
3. Ott, E.: Chaos in dynamical systems. Cambridge Univ. Press (1993)
4. Bose, B.K.: Neural network applications in power electronics and motor drives - an introduction and perspective. IEEE Trans. Ind. Electron. 54(1), 14–33 (2007)
5. Rodriguez, J., Rivera, M., Kolar, J.W., Wheeler, P.W.: A Review of Control and Modulation Methods for Matrix Converters. IEEE Trans. Ind. Electron. 59(1), 58–70 (2012)
6. Chua, L.O.: A nonlinear dynamics perspective of Wolfram's new kind of science, I, II. World Scientific (2005)
7. Rosin, P.L.: Training cellular automata for image processing. IEEE Trans. Image Process. 15(7), 2076–2087 (2006)
8. Wada, W., Kuroiwa, J., Nara, S.: Completely reproducible description of digital sound data with cellular automata. Physics Letters A 306, 110–115 (2002)
9. Torikai, H., Funew, A., Saito, T.: Digital spiking neuron and its learning for approximation of various spike-trains. Neural Networks 21, 140–149 (2008)
10. Iguchi, T., Hirata, A., Torikai, H.: Theoretical and heuristic synthesis of digital spiking neurons for spike-pattern-division multiplexing. IEICE Trans. Fundamentals, E93-A 8, 1486–1496 (2010)
11. Gray, D.L., Michel, A.N.: A training algorithm for binary feed forward neural networks. IEEE Trans. Neural Networks 3(2), 176–194 (1992)
12. Nakayama, Y., Ito, R., Saito, T.: A Simple Class of Binary Neural Networks and Logical Synthesis. IEICE Trans. Fundamentals, E94-A 9, 1586–1589 (2011)

Stock Price Prediction Based on a Network with Gaussian Kernel Functions

Dong Kyu Kim and Rhee Man Kil*

College of Information and Communication Engineering, Sungkyunkwan Univesity
2066, Seobu-ro, Jangan-gu, Suwon, Gyeonggi-do, 440-746, Korea
goldencrosser@gmail.com, rmkil@skku.edu

Abstract. This paper presents a new method of stock price prediction based on the phase space analysis for stock price series. For the prediction model, a network with Gaussian kernel functions is selected and this network is optimized by using a noise variance estimate. As a result, the proposed model provides the high accuracy of predicted values. Through the simulation for the prediction of KOSPI 200 stock price values, the effectiveness of the proposed prediction model has been demonstrated.

Keywords: stock price prediction, regression model, nonparametric estimation, noise variance.

1 Introduction

It has been known that stock price values are generated by a nonlinear and non-stationary dynamic system according to the trend of market which is associated with economic situation. For the problem of time series prediction, the linear regression models such as ARMA and ARIMA models are usually used. However, these models may not provide good estimates of predicted stock price values. If the stock price values are complete random, it is not possible to predict the future values. However, in many cases, the stock price values are not completely random. Rather, in most cases, they show coherent or sometimes chaotic behavior [1]. In this case, stock price values can be described by a nonlinear dynamics. In this respect, the nonlinear and nonparametric regression model is investigated to provide the better estimates of predicted values. As a nonlinear regression model, a network with Gaussian kernel functions is considered since this model is a nonparametric estimation model and good for incremental learning due to the locality of kernel functions. Here, the problems are to determine the input structure for the given prediction of stock price values and also to determine the proper size of regression models for the given stock price series. For this purpose, the input structure is investigated by analyzing the phase space of stock price dynamics and the proper size of regression models is investigated using the estimation of noise variances embedded in stock price series. As a result, the proposed model provides an accurate estimation of predicted values.

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 705–712, 2013.

To demonstrate the effectiveness of the proposed model, the Korea Composite Stock Price Index (KOSPI) 200 stock price series from June 2008 to June 2012 is used for the prediction of stock price values. The proposed model can also be applied to other cases of time series prediction since our model is not just restricted to the problem of stock price prediction.

2 Phase Space Analysis of Stock Price Series

The goal of time series prediction is to predict future values from a series of past values observed at regular time intervals. For the modeling of time series data, we usually rely on the theory called the delay coordinate embedding proposed by Packard et al. [2] and Takens [3]. Their purpose is to find the appropriate dimension of attractors generated by the dynamical system. To describe the dynamical system, let us assume that there exists a D dimensional state vector $\boldsymbol{x}(t)$, $t \in \mathbb{R}$; that is,

$$\boldsymbol{x}(t) = [x(t), x^{(1)}(t), \cdots, x^{(D-1)}(t)]^T \tag{1}$$

where $x^{(i)}(t)$ represents the ith derivative of $x(t)$ with respect to t. This vector can be produced by a dynamical system,

$$\dot{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{x}, t) \tag{2}$$

where $\boldsymbol{F}$ represents the D dimensional mapping function from the state vector $\boldsymbol{x}$ to the vector $\dot{\boldsymbol{x}}$ at time t. With no information on the state space (or phase space), we can only observe $x(t)$. Moreover, we usually consider the sampled discrete time series data $x(t_k)$; that is,

$$x(t_k) = x(t_0 + k\Delta t) \equiv x(k), \quad k = 0, 1, 2, \cdots \tag{3}$$

where t_0, k, and Δt represent the initial time, sampling step, and sampling interval, respectively. Here, let us construct E-dimensional vectors

$$\boldsymbol{x}_{\tau,E}(k) = [x(k), x(k-\tau), \ldots, x(k-(E-1)\tau)]^T, \quad k = (E-1)\tau, \cdots \tag{4}$$

where τ and E represent the delay time measured as the unit of Δt and embedding dimension, respectively.

For the prediction of future values, we assume that the state vectors in the reconstructed state space are governed by a nonlinear function f; that is,

$$x(t_k + P) = f(\boldsymbol{x}_{\tau,E}(k)) \tag{5}$$

where P represents the prediction step. Then, our goal is to make a prediction model $\hat{f}$ estimating the unknown target function f. However, since the geometric structure of f is determined by the embedding parameters τ and E, the performance of the prediction model is strongly affected by the choice of the embedding parameters. One of the typical methods is to choose the embedding dimension E by estimating the correlation dimension of the system [4], and to

choose the delay time τ at which the first minimum of the mutual information occurs. However, the correlation dimension does not indicate any information on the target function f. In this context, we consider to determine the embedding parameters using the measure related to the smoothness of the target function.

For the analysis of time series data, let us denote the rth nearest neighbor of the vector $\boldsymbol{x}_{\tau,E}(k)$ by $\boldsymbol{x}^r_{\tau,E}(k)$. Since the difficulty of estimation depends on the smoothness of the target function f, we define the gradient of f at each point $\boldsymbol{x}_{\tau,E}(k)$ by

$$\Delta f(\boldsymbol{x}^r_{\tau,E}) = \frac{\left|f(\boldsymbol{x}_{\tau,E}(k)) - f(\boldsymbol{x}^r_{\tau,E}(k))\right|}{||\boldsymbol{x}_{\tau,E}(k) - \boldsymbol{x}^r_{\tau,E}(k)||}, \tag{6}$$

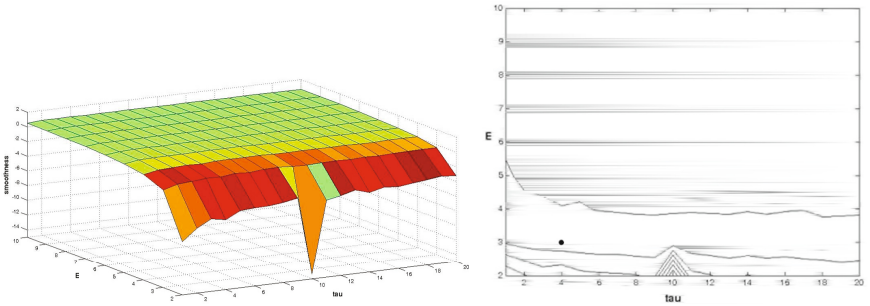
where the norm in the denominator represents the Euclidian distance in $\mathbb{R}^E$. In this paper, $r = 1$ is used; that is, the nearest neighbor of the state vector. Here, the smoothness measure $S(\tau, E)$ of a target function f [5] is defined by using an average of the gradient values for time series $x(t_k)$, $k = 0, 1, \cdots, n-1$; that is,

$$S(\tau, E) = 1 - \frac{1}{n - (E-1)\tau} \sum_{k=(E-1)\tau}^{n-1} \Delta f(\boldsymbol{x}^1_{\tau,E}(k)). \tag{7}$$

The smoothness measure $S(\tau, E)$ is closely related to the unfolding of the orbits of the vectors in the reconstructed state space. If the embedding dimension E is too small, the state space is yet restricted to the low dimensional space. This restriction makes different orbits of vectors to be folded and placed closely to each other; that is, even the nearest neighbor vectors might have come from different orbits. Since the vectors on the different orbits are likely to have largely different target function values, large gradient values imply that the embedding dimension is too small. On the other hand, if the embedding dimension E is large enough, the nearest neighbor vectors are obtained from the same orbit, which results in the small gradient values. Also, the smoothness measure $S(\tau, E)$ will not be much influenced by changes in E when E is large, because the state space is already unfolded. From these observations, the optimal embedding dimension and delay time can be determined by identifying the points where the smoothness measure $S(\tau, E)$ changes rapidly from the smaller values to the larger values. For the analysis of one step prediction for KOSPI 200 stock prices, the smoothness measure of (7) is calculated for every delay time τ between 1 and 20, and every embedding dimension E between 2 and 10. The calculated smoothness measure of KOSPI 200 stock prices is illustrated in Fig. 1. From the plot of smoothness measure, the smallest embedding dimension is selected first. Here, the proper embedding dimension is selected as $E = 3$. Then, for the given embedding dimension $E = 3$, the proper delay time is selected as $\tau = 4$. As a result, the model of one step prediction is determined as

$$x(t+1) = f(x(t), x(t-4), x(t-8)). \tag{8}$$

From this phase space analysis, it is evident that the information of 3 data within 8 day window provides an important cue to determine the next day prediction of KOSPI 200 stock price values.



(a) (b)

Fig. 1. The plot of smoothness measure for stock price prediction: (a) and (b) represent the 3D plot and contour map of stock price series. The dot in the contour map represents the selected delay time $\tau = 4$ and embedding dimension $E = 3$.

3 Nonlinear Stock Price Prediction Model

As a nonlinear prediction model for stock prices, a network with Gaussian kernel function is selected since this network is able to perform nonlinear and nonparametric estimation and good for incremental learning due to the locality of kernel functions. The suggested prediction model $\hat{f}$ with m kernel functions is described by

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{m} w_i \psi_i(\boldsymbol{x}), \quad \psi_i(\boldsymbol{x}) = e^{-||\boldsymbol{x}-\boldsymbol{\mu}_i||^2/2\sigma_i^2}, \tag{9}$$

where w_i represents the connection weight between the output and the ith kernel function ψ_i in which $\boldsymbol{\mu}_i$ and σ_i represent the mean and standard deviation, respectively. In (9), we need to determine the parameters of m and also kernel related parameters w_i, $\boldsymbol{\mu}_i$, σ_i. Foe kernel related parameters, [6] suggested an efficient estimation method in such a way of minimizing the mean square error (MSE). However, for our problem of prediction model, an optimal number m of kernel functions should be determined to be fitted to the stock price series so that the generalization error is minimized. Here, in the case of time series $x(t_k + P)$, it can be described by

$$x(t_k + P) = f(\boldsymbol{x}_{\tau,E}(k)) + \epsilon, \tag{10}$$

where f represent an embedded function for the generation of time series data and ϵ represents a noise term expressed as a random variable with mean 0 and variance σ^2.

Then, for $x(t_k + P)$, the predicted value $\hat{x}(t_k + P)$ is described by

$$\hat{x}(t_k + P) = \hat{f}(\boldsymbol{x}_{\tau,E}(k)). \tag{11}$$

The expected risk (or MSE) between the true and predicted values is described by

$$E[(x - \hat{x})^2] = E[(f - \hat{f})^2] + Var(\epsilon), \tag{12}$$

where $E[(f - \hat{f})^2]$ represents the regression error and $Var(\epsilon)$ represents the variance of noise.

From the above equation, it is clear that the expected risk is always greater than or equal to the variance of noise. This implies that the empirical error (or training error) should not be less than the variance of noise to avoid over-fitting of the regression model for the given time series. Here, the variance of noise should be estimated. One method of estimating the variance of noise is using the squares of time series differences [7]:

$$\hat{\sigma}^2 = \frac{1}{2(n-1)} \sum_{k=1}^{n-1} (x(t_k) - x(t_{k-1}))^2, \tag{13}$$

where $\hat{\sigma}^2$ represents an estimator of noise variance.

Assuming that the random noise follows a normal distribution, an estimator of (13) follows a chi-square distribution; that is,

$$\frac{(n-1)\hat{\sigma}^2}{\sigma^2} \sim \chi^2_{n-1}. \tag{14}$$

Then, with a confidence level of $1 - \alpha$, the following probability of (14) is described:

$$P\left(\chi^2_{1-\alpha/2,n-1} \leq \frac{(n-1)\hat{\sigma}^2}{\sigma^2} \leq \chi^2_{\alpha/2,n-1}\right) = 1 - \alpha. \tag{15}$$

From the above equation, a $100(1-\alpha)\%$ confidence interval for noise variance σ^2 is determined by

$$\frac{(n-1)\hat{\sigma}^2}{\chi^2_{\alpha/2,n-1}} \leq \sigma^2 \leq \frac{(n-1)\hat{\sigma}^2}{\chi^2_{1-\alpha/2,n-1}}. \tag{16}$$

In the case of large sample size, this interval is also effective even if the distribution of random noise is unknown because both an estimator of (13) and chi-square random variable approximately follow a normal distribution for large n according to the central limit theorem. Then, from (12) and (16), it is recommended that the training error should not be within the confidence interval for noise variance. From this point of view, the following algorithm of constructing a stock price prediction model is suggested:

Construction of Nonlinear Stock Price Prediction Model

Step 1. From the given stock price series $x(t_k)$, $k = 0, 1, \cdots, n-1$ and prediction time P, determine the values of smoothness measure of (7). Then, determine the embedding dimension E and delay time τ when the value of smoothness measure is large (usually, greater than -1) at the smaller embedding dimension.

Step 2. Determine the structure of nonlinear prediction model of (5).

Step 3. Estimate the noise variance using (13) and determine a $100(1-\alpha)\%$ confidence interval for noise variance using (16).

Step 4. Train the nonlinear prediction model using an incremental learning algorithm such as [6] as the number of kernel functions increases and determine the following training error:

$$R_{emp}(\hat{f}) = \frac{1}{n}\sum_{k=0}^{n-1}(x(t_k) - \hat{x}(t_k))^2. \tag{17}$$

Step 5. If the condition

$$R_{emp}(\hat{f}) > \frac{(n-1)\hat{\sigma}^2}{\chi^2_{1-\alpha/2,n-1}} \tag{18}$$

is met, continue the learning process of Step 4. Otherwise, stop the learning process.

This construction algorithm was applied to KOSPI 200 stock price series using the prediction model of (8). In this experiment, an 80% of stock price series was used as training data and the remaining 20% was used as test data. Here, the noise variance was estimated as $\hat{\sigma}^2 = 0.000127$ and a 99% confidence interval for noise variance σ^2 was determined by

$$0.000115 \leq \sigma^2 \leq 0.000159.$$

Here, to compare with the training and test error with respect to the number of kernel functions, the training error, test error, and confidence interval were plotted as illustrated in Fig. 2.

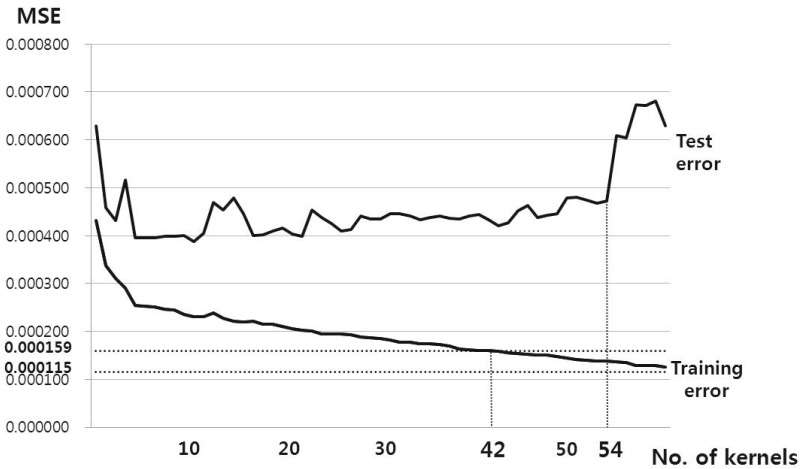


Fig. 2. Selection of the number of kernel functions by comparing the training error with the confidence interval for noise variance

As shown in Fig. 2, the over-fitting of regression model occurs when the number of kernel function is greater than or equal to 54. In this case, the training error is within the confidence interval for noise variance. Therefore, in this

experiment, it is clear that the proper number of kernel functions is less than or equal to 42 which makes the training error just greater than the upper limit of the confidence interval.

4 Simulation

For the simulation for stock price prediction, the data set of Korea Composite Stock Price Index (KOSPI) 200 stock price series from June 2008 to June 2012 was selected and normalized by dividing the price values by 200 so that approximate price range was between 0.5 and 1.5. In this data set, the first 70%, 80%, and 90% of data were used as training data sets, and the remaining data sets were used as test data sets. First, to identify the dynamics of stock price series for the given prediction time $P = 1$, the embedding dimension E and delay time τ were determined using the smoothness measure of (7). As a result, the prediction model of (8) was obtained. Then, the noise variance was estimated using the estimator of (13) and a 99% confidence interval for noise variance was determined using the form of (16). For the given prediction model, the parameters of Gaussian kernel function network (GKFN) of (9) were trained using the nonparametric estimation method of [6] as the number of kernel functions increased. This training was continuously performed until the training error reached the upper limit of the confidence interval for noise variance. Here, to measure the performances of stock price prediction, the following root mean square error (RMSE) and the coefficient of determination R^2 were use for the test data $y(t_i)$, $i = 0, \cdots, l-1$:

$$RMSE = \left(\frac{1}{l} \sum_{i=0}^{l-1} (y(t_i) - \hat{y}(t_i))^2 \right)^{1/2}, \tag{19}$$

where l represents the number of test data and

$$R^2 = 1 - \left(\sum_{l=0}^{l-1} (y(t_i) - \hat{y}(t_i))^2 / \sum_{l=0}^{l-1} (y(t_i) - \bar{y})^2 \right), \tag{20}$$

where $\bar{y}$ represent the sample mean of $y(t_i)$; that is, $\bar{y} = (1/l) \sum_{l=0}^{l-1} y(t_i)$. Here, the range of R^2 is between 0 and 1, and R^2 indicates the degree of how well the regression model covers the variation of time series data; for example, if $R^2 = 0.9$, the regression model covers 90% of the variation of time series data and the remaining 10% of the variation is not explained. Then, the simulation results for stock price prediction using the proposed GKFN were obtained and described in Table 1. For the comparison, simulation results for stock price prediction using the kernel support vector machine (k-SVM) [8], one of popular methods in nonlinear regression models, were also obtained. In the k-SVM, the same form of the prediction model of (8) was also used.

These simulation results showed that the RMSE was decreased as the ratio of training data was increased and the proposed GKFN outperformed the k-SVM from the view points of the RMSE and also R^2. In general, if R^2 is greater

Table 1. Simulation results for stock price prediction

Ratio of training data	k-SVM		GKFN	
	RMSE	R^2	RMSE	R^2
70%	0.058423	0.60	**0.035879**	**0.87**
80%	0.039960	0.64	**0.020517**	**0.90**
90%	0.036820	0.41	**0.016172**	**0.88**

than or equal to 0.9, we consider that the regression model performs very good fit to the given data. The proposed GKFN was very close to this performance. The main reason was due to the fact that the prediction model was determined by phase space analysis and the optimization of regression model was done by comparing the training error with the noise variance estimate.

5 Conclusion

The stock price prediction involves the analysis of stock price series and also optimization of regression models. In this work, the stock price series is analyzed by the phase space analysis method [5]. As the prediction model, a network with Gaussian kernel functions is selected since this model is good for incremental learning due to the locality of kernel functions. For the optimization of regression model, the noise variance is estimated and used to determine the proper number of kernel functions. As a result, the proposed model provides the high accuracy of predicted values. Through the simulation for the prediction of KOSPI 200 stock price values, the effectiveness of the proposed prediction model has been demonstrated. The proposed model can also be applied to various problems of time series prediction.

References

1. Peters, E.E.: Fractal market analysis: applying chaos theory to investment and economics. John Wiley & Sons, New York (1994)
2. Packard, N.H., Crutchfield, J.P., Farmer, J.D., Shaw, R.S.: Geometry from a Time Series. Phys. Rev. Lett. 45, 712–716 (1980)
3. Takens, F.: Detecting strange attractors in turbulence. In: Rand, D.A., Young, L.S. (eds.) EAMT-WS 1993. Lecture Notes in Mathematics, vol. 898, pp. 366–381. Springer-Verlag, Berlin (1981)
4. Havstad, J.W., Ehlers, C.L.: Attractor dimension of nonstationary dynamical systems from small data sets. Phys. Rev. A 39(2), 845–853 (1989)
5. Kil, R., Park, S., Kim, S.: Time series analysis based on the smoothness measure of mapping in the phse space of attractors. In: International Joint Conference on Neural Networks, vol. 4, pp. 2584–2589 (1999)
6. Kil, R.: Function approximation based on a network with kernel functions of bounds and locality. ETRI Journal 15, 35–51 (1993)
7. Rice, J.: Bandwidth choice for nonparametric regression. Annals of Statistics 12(4), 1215–1230 (1984)
8. SVMlight ver 6.02, Cornell University, `http://svmlight.joachims.org`

Enhanced GPU Accelerated K-Means Algorithm for Gene Clustering Based on a Merging Thread Strategy

Yau-King Lam, Peter W.M. Tsang, and Chi-Sing Leung

Dept. of Electronic Engineering, City University of Hong Kong, Hong Kong
kinglam4-c@my.cityu.edu.hk, {eewmtsan,eeleungc}@cityu.edu.hk

Abstract. Past research has demonstrated that gene clustering can be effectively accomplished with the K-means algorithm. Furthermore, the clustering process can be conducted swiftly with the use of graphic processing units (GPUs). However, due to the limited number of processing units (cores) on a GPU, the computation time will be lengthened if the number of gene data is too large. To alleviate this problem, a novel method for realizing the K-means algorithm on GPUs has been developed and presented in this paper. Essentially, a fragment shader program is implemented to process multiple data points in a single thread. Experimental results show that our proposed GPU accelerated scheme can attain over 25 % increase in the computation speed as compared with the existing method.

Keywords: Gene Clustering, K-Means, Merging Thread, GPU.

1 Introduction

The advancement of computing and the Microarray technologies [1][2] have enabled the extraction of important information from massive gene data sets. Amongst different methods, the cluster analysis is an important means to encapsulate the huge data sets into a relatively small group of characteristic features [3]. The latter is generally obtained with the K-means algorithm [4][5], a classical clustering tool which has been applied in numerous engineering and scientific disciplines.

Starting with an initial set of centroids, each of which corresponds to a characteristic feature, clustering of the data set to the group (a.k.a centroid) is conducted in repetitively rounds of calculations. At each iteration, each member in the data set (hereafter refer as a training vector) is allocated to the nearest centroid according to the Euclidean distance. The latter is then updated by averaging the values of the training vectors under its coverage. Despite the success of the K-means algorithm, the computation time is often overwhelming, especially in the processing of large data sets. This problem is particular serious in gene data clustering as both the size of the data set, and the dimension of the data points, are large.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 713–720, 2013.

Recently, graphics processing units (GPUs) have been rapidly evolved because of the requirements from computer game industries. They are also suitable for general purpose computations, such as image-based relighting [6], wavelet transform [7], neural network simulation [8,9], and pattern recognition [10].

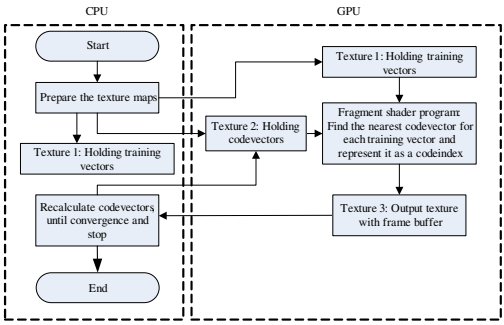


Fig. 1. Realization of the K-means algorithm with the GPU

Some GPU-based training realizations for K-means have also been proposed [11,12]. Takizawa et al. [11] proposed a CPU–GPU co-processing solution for K-means, where GPU is mainly for Euclidean distance calculation and CPU is for updating the centroids. In their implementation, a training iteration consists of a number of rendering passes, where each rendering pass is associated with a centroid. Within a rendering pass, a fragment shader is used for Euclidean distance calculation. Each training vector is associated with a fragment, and its temporary nearest centroid index and its corresponding distance are updated as the color values of the fragment. With the z-buffering functionality of GPU, the nearest centroid indices are available on the GPU when all the rendering passes are finished. The indices are then transferred to the main memory, and the CPU is account for sequentially updating the centroids. By using their GPU approach, the training speed can be accelerated by about 35 times, compared with a decent CPU implementation.

In the ideal single instruction multiple data (SIMD) situation, the determination of the nearest centroid for each training vector will be encapsulated as a single thread that is handled by a unique core processor. As such, the evaluation of the entire training data set in each iteration could be conducted in a totally parallel fashion. However, in practice, this ideality is not realizable as the number of core processors on a GPU is limited in number, and each of them has to operate on multiple threads in a sequential manner. In this paper, we report a method to alleviate this problem with a merging thread strategy, so that the evaluation of 2 or more training vectors can be combined into a single thread.

Organization of this paper is listed as follows. In Section 2, a brief review of the realization of the K-means algorithm on GPUs based on a classical fragment shader architecture is outlined. Subsequently, we shall describe our proposed

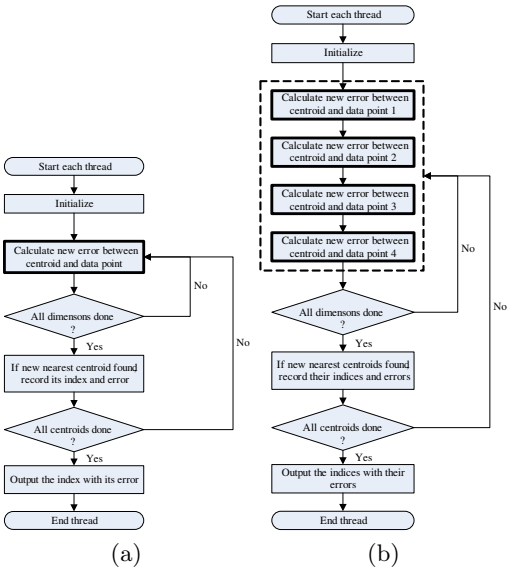


Fig. 2. The difference between our proposed method and the direct method. (a) The direct method. (b) Our proposed merging thread strategy.

method that incorporates the merging thread strategy into the fragment shader program. Experimental evaluation is given in Section 3, followed by a conclusion in Section 4.

2 Typical GPU Realization of the K-Means Clustering Algorithm

In the Takizawa's approach [11], a training iteration consists of a number of rendering passes. At each rendering pass, a fragment shader is used for calculating the distance from the centroid to each training vector and for comparing the calculated distance with the temporary minimum distance. The reason of using multiple passes is that at their time the number of instructions in a shader is limited. Nowadays, due to the improvement of the GPU technology, a single rendering pass is able to determine the nearest centroid.

A general GPU realization is shown in Fig. 1. The CPU first prepares the training vectors and initial centroids as textures. Those textures are then stored on the GPU. On the GPU, a fragment shader determines the nearest centroid for each training vector in a parallel manner, and the nearest centroid index of each training vector is returned to the CPU. The mean square error (MSE) of the quantization of the training vector is also returned to the CPU. The CPU then calculates the new set of centroids and the average MSE. The updated centroids are passed to the fragment shader program on the GPU, and invokes next round

of clustering. The process repeats until the average MSE is lower than certain threshold, or when the number of iterations exceeds a given limit.

Fig. 2(a) shows the implementation for the fragment shader program with the direct method. In this direct method, for each training vector, the fragment shader calculate the distances from the training vector to all centroids. After the calculation, the nearest centroid index and the MSE of the quantization are obtained. From the parallel computing's point of view, the evaluation of the nearest centroid for each training vector is treated as a thread. Since there are a lot of training vectors, many threads are created. In the ideal case, each thread will be handled independently by a single processor. However, in practice, this ideality is not realizable as the number of core processors on a GPU is limited in number. Hence each GPU core processor has to operate on multiple threads in a sequential manner. Also, those threads compete for the same set of resources, such as GPU memory access. In view of this, we propose a merging thread strategy to reduce the competition between those threads.

3 Merging Thread Strategy

We propose a merging thread strategy so that a plurality of training vectors will be grouped into a thread before they are evaluated by their associated core processor. In this merging thread approach, a thread processes a few training vectors. The new fragment shader program for processing four training vectors is shown in Fig. 2(b). The corresponding Cg codes is shown in Listing 1.1.

Without loss of generality, we consider that each thread processes four training vectors. There are two input textures for this merging thread strategy. One texture, calling *codebook texture*, stores the centroids. Another texture, calling *training vector texture*, stores the training vectors. Let k be the number elements in the data vectors, let M be the number of centroids, and let N be the number of training vectors. As a texel can store four floating-point values (RGBA), an k-dimensional vector occupies $\frac{k}{4}$ texels. Therefore, the resolution of the centriod texture is equal to $M \times \frac{k}{4}$, as shown in Fig. 3(a). The resolution of the training

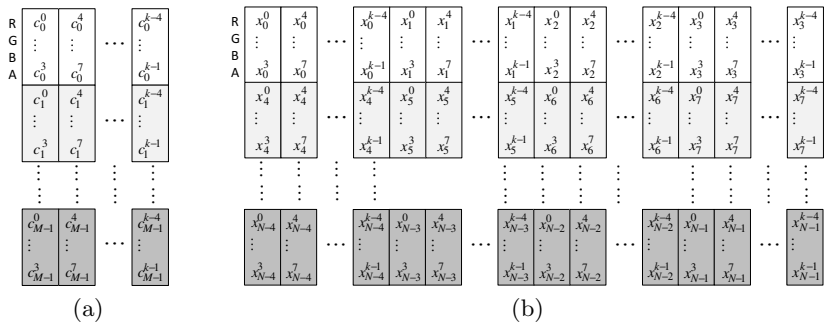


Fig. 3. The organization of textures. (a) Codebook texture. (b) Training vector texture.

Listing 1.1. Our proposed method that incorporates the merging thread strategy (for 4 data points as an example) to enhance the computation efficiency of the fragment shader in realizing the K-means algorithm

```
#define FLT_MAX 3.402823466e+38F
#define P_BookSize  256 //the size of codebook
#define P_TexDim    (256/4)  //the number of dimensions in unit of texel
//if new min. distance, then record the new one
#define IF_MIN_REC(f2Best, dist, k)    if ( (dist)<f2Best.y ) {(f2Best) =
      float2((k), (dist)); }
//evaluate the distance for each data point
#define EVALUATE_DIST(dist1)    \
  t4 = p4 - texRECT( texData, f2pos );
  (dist1) += dot(t4,t4);
  f2pos.x += nOffset;
void main_FragKM_matching(
  float2 index: TEXCOORD0,        // The index of a vector in the data
      set
  uniform samplerRECT texData,    // Texture holding the data vectors
  uniform samplerRECT texBook,    // Texture holding the codebook (
      centroids)
  out float4 oResults: COLOR0,    // Textures holding the output (code
      indices)
  out float4 oResults1: COLOR1 ){
  float4 p4, t4;
  int i, j, k, nOffset;
  float xStart;
  float fSSE;
  float2 f2pos;
  float dist1, dist2, dist3, dist4;// the minimum error with
      corresponding
                                   //index for the data vectors
  float2 out1, out2, out3, out4;
  out4 = out3 = out2 = out1 = float2(-1, FLT_MAX);
  xStart = (index.x-0.5)*P_TexDim+0.5;
  f2pos.y = index.y;
  nOffset = (P_QuadWidth*P_TexDim);

  for (k=0; k<P_BookSize; k++) {
  //reset the distances for the data vectors
    dist4 = dist3 = dist2 = dist1 = 0;
    for ( i=0; i<P_TexDim; i++ ) {
      // share the centrods for more data vectors at a time
      p4 = texRECT( texBook, float2(i+0.5, k+0.5) );
      //the k-th vector in codebook
      f2pos.x = (xStart+i);
      EVALUATE_DIST(dist1);  // the 1st data vector
      EVALUATE_DIST(dist2);  // the 2nd data vector
      EVALUATE_DIST(dist3);  // the 3rd data vector
      EVALUATE_DIST(dist4);  // the 4th data vector
    } // i<P_TexDim for a vector of all dimensions
    IF_MIN_REC(out1, dist1, k)
    IF_MIN_REC(out2, dist2, k)
    IF_MIN_REC(out3, dist3, k)
    IF_MIN_REC(out4, dist4, k)
  } //for k<P_BookSize
  //sum the squared error of the results
  fSSE = out1.y + out2.y + out3.y + out4.y;
  // output texels for the fragment in the two textures
  oResults = float4(out1.x, out2.x, out3.x, out4.x);
  oResults1 = float4(fSSE, 0, 0, 0);
}
```

vector texture is $\frac{N}{4} \times k$, as shown in Fig. 3(b). With the above arrangement, each row of the training vector texture corresponds to four training vectors. After preparing these two textures, they are put into the GPU memory. We also define two output textures. One is used to store the nearest centroid indices, and its resolution [1] is equal to $1 \times \frac{N}{4}$. It is called as *centroid index texture*. Another output texture, calling *MSE texture*, is used to hold the MSE values of the quantization, and its resolution is also equal to $1 \times \frac{N}{4}$.

To start the fragment shader, in the OpenGL environment, we draw a picture with resolution of $1 \times \frac{N}{4}$. That means, we define $\frac{N}{4}$ fragments, as well as $\frac{N}{4}$ threads, on the GPU. The texture coordinate of each fragment is used to indicate the four training vectors that the fragment owns. For each fragment, the fragment shader then find out the corresponding four nearest centroids. Afterwards, the nearest centroid indices and MSE values are output to the centroid index texture and the MSE texture, respectively. Thereafter, the CPU updates the codebook. The updated codebook is then passed to the GPU for the next round of clustering.

4 Experimental Evaluation and Discussion

The proposed scheme is evaluated with two popular gene datasets. The Lymphoma [13] gene expression dataset consists of 4022 data. Each of them is represented by 96 dimensional vector. The dataset is partitioned into 256 clusters.

The Yeast cell-cycle [14] dataset is comprising of 6178 data. Each of them is represented as a 77 dimensional vector. The dataset is partitioned into 256 clusters with the K-means algorithm running on the GPU.

All the evaluations are conducted on the CPU equipped with "Intel Core i7 920" CPU, and the "nVidia GTX260" GPU card. We compare the realization of the K-means algorithm using the direct method in Fig. 2(a), that employs the traditional fragment shader, and our proposed method in Fig.2(b).

The performance (in terms of number of iterations per second) of the direct method, and our proposed method based on the merging of 2 to 4 training vectors, is shown in Table 1. The results are the average values of 10 runs. We observed that the proposed merging thread strategy could result in over 25% improvement in the computation efficiency when we merge four training vectors into one thread.

There are several reasons why the merging thread strategy can improve the efficient. First, using the merging thread strategy can reduce the overhead for switching threads. In the direct method, the number of threads is equal to the number N of training vectors. The merging thread strategy can reduce the number of threads,as well as the overhead for switching threads. Second, in the direct method and the merging thread strategy, each thread needs to access the whole codebook one time. Therefore, at each training iteration, the direct method accesses the whole codebook N times. On the other hand, the merging thread strategy accesses the whole codebook $\frac{N}{4}$ times only.

[1] Note that each texel can hold four values.

Table 1. Speed, and speed-up gain of of our proposed scheme with respect to the direct method for gene clustering

Data Sets	The number of merging threads	Speed	Speed up gain
Lymphoma	1 (direct method)	87.3	0
	2	105.2	21%
	3	109.8	26%
	4	109.8	26%
Yeast-cell-cycle	1 (direct method)	69.5	0
	2	79.3	14%
	3	83.7	20%
	4	87.8	26%

5 Conclusion

In this paper we reported an enhanced method for realizing gene clustering on a GPU. We have built a fragment shader program to conduct the K-means algorithm, which is employed to partition the gene data into a group of centroids. A merging thread strategy for reducing the number of threads associated with each processing core, has been adopted to increase the computational efficiency. Experimental evaluation reveals that our proposed method is capable of increasing the computation speed by over 25% as compare with existing approach.

Acknowledgement. The work was supported by RGC General Research Fund from Hong Kong (Project No.: CityU 116511).

References

1. Brown, P., Botstein, D.: Exploring the new world of the genome with dna microarrays. Nature Genetics 21, 33–37 (1999)
2. Brazma, A., Robinson, A., Cameron, G., Ashburner, M.: One-stop shop for microarray data. Nature 403, 699–700 (2000)
3. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences of the United States of America 95, 14863–14868 (1998)
4. Hartigan, J.A., Wong, M.A.: A k-means clustering algorithm. Appl. Stat. 28, 126–130 (1979)
5. Shamir, R., Sharan, R.: Algorithmic approaches to clustering gene expression data. In: Jiang, J., Smith, T., Xu, Y., Zhang, M. (eds.) Current Topics in Computational Biology, pp. 269–299. MIT Press, Cambridge (2002)
6. Leung, C.S., Wong, T.T., Lam, P.M., Choy, K.H.: An RBF-based image compression method for image-based rendering. IEEE Trans. on Image Processing 15(1), 1031–1041 (2006)

7. Wong, T.T., Leung, C.S., Heng, P.A., Wang, J.: Discrete wavelet transform on consumer-level graphics hardware. IEEE Trans. on Multimedia 9(3), 668–673 (2007)
8. Ho, T.Y., Lam, P.M., Leung, C.S.: Parallelization of cellular neural networks on GPU. Pattern Recogn. 41(8), 2684–2692 (2008)
9. Bohn, C.A.: Kohonen feature mapping through graphics hardware. In: Proceedings of 3rd Int. Conference on Computational Intelligence and Neurosciences, pp. 64–67 (1998)
10. Leung, C.S., Lam, P.M., Tsang, P.W.M., Situ, W.: A graphics processing unit accelerated genetic algorithm for affine invariant matching of broken contours. Journal of Signal Processing Systems 66(2), 105–111 (2012)
11. Takizawa, H., Kobayashi, K.: Hierarchical parallel processing of large scale data clustering on a PC cluster with GPU co-processing. J. Supercomput. 36, 219–234 (2006)
12. Lam, Y.-K., Tsang, P.W.M., Leung, C.-S.: Improved gene clustering based on particle swarm optimization, k-means, and cluster matching. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part I. LNCS, vol. 7062, pp. 654–661. Springer, Heidelberg (2011)
13. Alizadeh, A.A., Eisen, M.B., Davis, R.E., Ma, C., Lossos, I.S., Rosenwald, A., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., Powell, J.I., Yang, L., Marti, G.E., Moore, T., Hudson, J.J., Lu, L., Lewis, D.B., Tibshirani, R., Sherlock, G., Chan, W.C., Greiner, T.C., Weisenburger, D.D., Armitage, J.O., Warnke, R., Levy, R., Wilson, W., Grever, M.R., Byrd, J.C., Botstein, D., Brown, P.O., Staudt, L.M.: Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
14. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast, saccharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell 9, 3273–3297 (1998)

Multimodal Feature Learning for Gait Biometric Based Human Identity Recognition

Emdad Hossain and Girija Chetty

IT&E, Faculty of ESTeM,
University of Canberra, Australia
{emdad.hossain,girija.chetty}@canberra.edu.au

Abstract. In this paper we propose a novel multimodal feature learning technique based on deep learning for gait biometric based human-identification scheme from surveillance videos. Experimental evaluation of proposed learning features based on novel deep learning and standard (PCA/LDA) features in combination with classifier techniques (NN/MLP/SVM/SMO) on different datasets from two gait databases (the publicly available CASIA multiview multispectral database, and the UCMG multiview database), show a significant improvement in recognition accuracies with proposed fused deep learning features.

Keywords: multimodal, multiview, gait, vPCA, Deep Learning, identification, fusion.

1 Introduction

Over the last few years, several research works have been reported on use of different biometric modalities for establishing identity, and have recognized the importance of gait patterns (walking style), for recognizing human ID at a distance, where the face cannot be clearly seen in video footage. However, these complex surveillance environments are characterized by uncooperative cameras and uncooperative subjects, where most of traditional approaches used for biometric identity recognition with face, iris and fingerprint biometrics normally fail. However, several physiological and biomechanical studies have shown that human gait is a unique and an inherently multimodal biometric, and involves a complex kinematic interaction between several motion articulators, and includes interplay between lower and upper limbs and other biomechanics of joints. It is person specific based on body weight, height, joint mobility in the limbs, and other person specific behavioral nuances. Due to this it is unique and cannot be forged, and if we can model these inherently multimodal traits by extracting compact representations from multiple sources, in terms of robust features and combine them appropriately, it is possible to identify humans from a distance from their gait or from the way they walk, irrespective of uncooperative cameras and uncooperative subjects. Automatic identification systems built using these features, can make a great contribution to surveillance and security area, can lead to better understanding of gait abnormalities, and lead to development of better human

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 721–728, 2013.

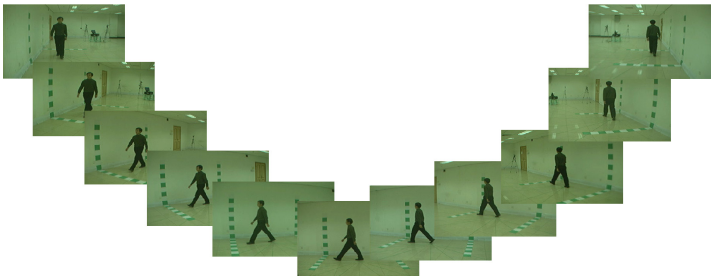
computer interfaces. However, each of these cues or traits captured from long range low resolution surveillance video cameras on their own are not powerful enough for ascertaining identity, a combination or fusion of each of them, along with suitable automatic processing approaches can result in robust recognition.

In this paper, we propose usage of full profile silhouettes of persons from multi-view low resolution cameras for capturing inherently multi-modal cues available from the gait patterns of the walking humans, and use it for establishing their identity. Further, we propose the use of unsupervised feature learning techniques, based on variants of principal component analysis (PCA) and deep learning (DL) approaches, which allow an in-depth analysis of the underlying pixel data. We compare these new features to standard features based on multivariate statistical techniques, such as PCA and linear discriminant analysis (LDA), along with well known learning classifier approaches based on support vector machines, NN and MLP classifiers [2, 19, 27]. The experimental evaluation of the proposed approach with two different databases, the publicly available CASIA [1] gait database, and the newly developed UCMG database [22], show a significant improvement in recognition performance with the proposed unsupervised learning features, as compared to standard features proposed in the literature, particularly for uncooperative camera conditions, simulated with mismatched train and test data sets. The rest of the paper is organised as follows. Next Section describes the background and motivation for proposed work, followed by the proposed multiview multimodal feature learning scheme in Section 3. The details of the experiments performed are described in Section 4, and conclusions and plans for further work are described in Section 5.

2 Background

To address the next generation security and surveillance requirements for not just high security environments, but also for day-to-day civilian access control applications with low level security requirements, we need a robust and invariant biometric trait [3]. According to the authors in [4], the expectations of next generation identity verification involve, addressing issues related to application requirements, user concern and integration. Some of the suggestions made to address the requirements of these emerging applications were use of non-intrusive biometric traits, role of soft biometrics or dominant primary and non-dominant secondary identifiers and importance of novel automatic processing techniques. To conform to these recommendations; often there is a need to combine multiple physiological and behavioral biometric cues, leading to so called multimodal biometric identification system. While most behavioral biometrics are not unique enough to provide reliable human identification they have been proved to be sufficiently high accurate [5, 6]. Gait, is a similar powerful behavioral biometric, but as a single mode, on its own, it cannot be considered as a strong biometric to identify a person. However, if we combine complementary gait information from another source, the multi-modal combination is expected to be powerful for human identification. Researchers have found that one of the most promising techniques is the use of multimodality or combination of different biometric traits or same biometric trait from multiple

disparate sources. For example, researchers in [7, 8] have found that multi-modal scheme involving PCA on combined image of ear and face biometric results in significant improvement over either individual biometric. In addition, other recent attempts to improve the recognition accuracy include face, fingerprint and hand geometry [9]; face, fingerprint and speech [10]; face and iris [11]; face and ear [12]; and face and speech [13].



(a): CASIA database images

(b) UCMG database images

Fig. 1. Sample images from CASIA and UCMG gait databases

However, the power of automatic discovery of suitable feature representations extracted from disparate but complementary sources straight from pixels, that do not rely on elaborate computation intensive features and application-specific expert knowledge, and their fusion did not attract much attention from the research community. As opposed to traditional sophisticated computation intensive feature extraction stages, and use of domain specific expert knowledge to manually specify features, the proposed feature learning and discovery based on the variants of principal component analysis and the deep learning approach, seeks to optimize an objective function that captures the appropriateness of the features, and includes approaches based on energy minimization, manifold learning, and deep learning using auto-encoders [23, 24, 25].

3 Multimodal Feature Learning Scheme

For experimental evaluation of our proposed multimodal gait identification scheme, we used two different gait databases, CASIA Gait Database [1] and UCMG Gait databas3 [22]. Both the databases are large multi-view databases, and consist of video sequences of walking persons captured from multiple video cameras. Further, both the databases consist of subsets of gait data captured from multiple view angles (0, 30,60, 90 etc), with different walking styles (slow, normal, fast), and with different props (bag, hat, coat etc.). For all experiments reported in this paper, we used 10 subjects with a set of extracted silhouettes from Dataset B in CASIA and a dataset with similar matched conditions from the UCMG dataset. Each subject consists of 16 images and in total 160 images for 10 subjects (people). Figure 1 shows some sample sequences from CASIA Dataset B and UCMG Database.

3.1 Variant of PCA Features (*v-PCA*)

PCA is a basic form of feature learning and it allows automatic discovery of compact and meaningful representation of raw data without relying complex feature extractio techniques or on domain specific (or expert) knowledge. It is a well established technique used for decorrelation and dimensionality reduction of data. The variance of the original data is concentrated in low dimensional subspace characterized by eigenvectors and eigenvalues. The projection of the original data onto the variance-maximizing sub-space serves as a feature representation, and automatic analysis of the eigenvalue spectrum of the sample covariance uncovers the appropriate target-dimensionality of the feature space. However, the PCA features perform poorly if the input data are not properly normalized. Using blind range normalization does not solve the problem especially when the components relate to completely different aspects of a phenomenon. In the context of gait recognition from multiple views this becomes problematic and to address this issue we developed an alternate representation based on the empirical cumulative distribution function (ECDF) of the gait silhouette/contour (x, y) from each frame. This representation is independent of the absolute ranges but preserves structural information.

3.2 Deep Learning Features (DLF)

Hinton et al., [23] have proposed a powerful tool for generic semi-supervised discovery of features called *autoencoder networks*, which aim to learn a lower-dimensional representation of input data. This produces a minimal error when used for reconstructing the original data. For autoencoder based feature learning on sequential data we used a novel deep learning approach. In this approach, the lower dimensional features are discovered by means of feed-forward neural networks that consist of one input layer, one output layer and an odd number of hidden layers. Every layer is fully connected to the adjacent layers and a non-linear activation function is used. The objective function during training is the reconstruction of the input data at the output layer. The autoencoder transmits a description of the

input-data across each layer of the network. This non-linear low-dimensional encoding is hence an automatically learned feature representation. For robust model training, we used the techniques suggested by Hinton et al[24], where the layers of the autoencoder network are learnt greedily in a bottom-up procedure, by treating each pair of subsequent layers in the encoder as a Restricted Boltzmann Machine (RBM).

An RBM is a fully connected, bipartite, two-layer graphical model, which is able to generatively model data. It trains a set of stochastic binary hidden units which effectively act as low-level feature detectors. One RBM is trained for each pair of subsequent layers by treating the activation probabilities of the feature detectors of one RBM as input-data for the next. Once the stack of RBMs is trained, the generative model is unrolled to obtain our final fully initialized autoencoder network for feature learning. Different methods exist to model real-valued input units in RBMs. We used Gaussian visible units for the first level RBM that activate binary, stochastic feature detectors (Gaussian-binary). The subsequent layers rely on the common binary-binary RBM, and the final layer is a binary linear RBM, which effectively performs a linear projection. During training, the sample data is processed batch-wise, where each batch ideally comprises samples from all classes in the training-set. As the availability of the class information is not mandatory, we trained RBMs in a completely unsupervised manner. To evaluate the two feature learning approaches, *vPCA* and *DLF* features for gait based human identity recognition, we conducted a number of experiments using different subsets of data from the CASIA dataset B and UCMG database. For baseline comparison, we also extract standard PCA and LDA features. Further, we examined how these features perform with different classifiers and hence tested with Nearest Neighbor (NN), MLP, SVM and SMO classifiers, and is described in detail in some of the previously reported works [15, 16, 17, 19, 24, 27].

4 Experiments and Results

Three sets of experiments were performed on different subsets of data from two databases, the CASIA Dataset B (Visible spectrum), Dataset C(Infrared spectrum) and the. UCMG database. Table 1 to Table 5 show the recognition performance for each set of experiments in terms of recognition accuracy and several statistically significant performance measures such as true positive rate (TPR), false positive rate (FPR), precision, recall and Fmeasure.

By using PCA, LDA, vPCA and DLF feature learning techniques, we extracted the gait feature vectors from the silhouette images of walking humans in each video sequence, and performed identification experiments in single mode and multimodal fusion mode. To examine the performance of features under uncooperative camera conditions, we used different views for training and test conditions. We obtained a fused(averaged) training template by combining features extracted from different views, and used the testing data from a view other than those used for building the fused training template. Without this approach, the error becomes too large if training data is used from one view and test data is used from a different view.

Table 1. Performance of Learning Features (dimension = 20) with MLP classifier with 5 fold cross validation

Dataset	CASIA DB	UCMG DB
Features	Acc(%)	Acc(%)
PCA	48.63	46.62
vPCA	52.68	51.57
DLF	79.5	78.4
LDA	75.3	72.43

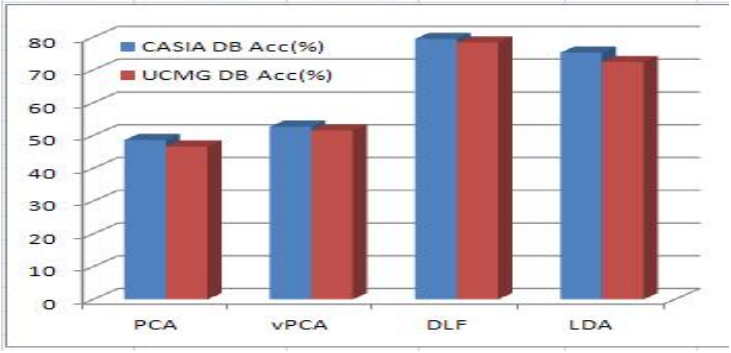


Table 2. Performance of Learning Features (dimension = 20) for different classifiers with 5 fold cross validation

Classifier	Database	Accuracy(%)	TP	FP	Precision	Recall	F-measure
NN	CASIA	92.5	0.93	0	0.93	0.93	0.93
NN	UCMG	92.25	0.92	0	0.93	0.92	0.92
SVM -L	CASIA	81.13	0.81	0	0.82	0.81	0.81
SVM -L	UCMG	78.75	0.78	0	0.81	0.79	0.78
SVM -RBF	CASIA	31.30%	0.3	0.02	0.74	0.3	0.39
SVM -poly	CASIA	27.63%	0.28	0.02	0.75	0.28	0.36
SVM -sigmoid	CASIA	29.13%	0.29	0.02	0.74	0.29	0.38

Table 3. Performance of Decision Fusion of LDF and DLF Features (dimension = 20) for CASIA Dataset B with different classifiers

Classifier	Database	Accuracy(%)	TPR	FPR	Precision	Recall	F-measure
NN	CASIA	98.38%	0.98	0	0.99	0.98	0.98
MLP	CASIA	74.88%	0.79	0.01	0.75	0.75	0.97
SVM-L	CASIA	98.25%	0.98	0	0.98	0.98	0.98
NN	UCMG	98.50%	0.99	0	0.99	0.99	1
MLP	UCMG	72.50%	0.73	0.01	0.77	0.73	0.73
SVM-L	UCMG	97.75%	0.98	0	0.98	0.98	0.98

The first set of experiments show the performance for proposed learning features in single mode for CASIA dataset B and UCMG datasets, under matched conditions (gender, walking styles and presence of props). For all experiments we used a reduced dataset, with 10 classes (10 persons) with 20 features (PCA, vPCA, LDA, DLF), as preliminary experimentation showed that about 95% of variations can be modelled by around 10 features, and any more increase in dimensionality does not result in significant improvement in performance. As can be seen in Table 1, the proposed deep learning and vPCA features perform better than the standard PCA features, though LDA features perform equally well here. The second set of experiments involved testing whether other established classifiers with different kernels result in better performance, and is shown in Table 2. As can be seen in Table 2, for both datasets, simple nearest neighbour classifier outperforms other sophisticated SVM classifier for all kernel types, which could be due to better learning ability of DLF features. The final set of experiments involved examining the decision level fusion of LDA and DLF features for different types of classifiers. As can be seen in Table 3, the decision fusion of LDA and DLF features results is better performance than the single mode features, particularly for NN and SVM classifier with linear kernels.

5 Conclusions

In this paper we proposed novel learning features based on deep learning and variants of PCA from long range gait profiles in surveillance videos. We investigated the performance of these features for uncooperative camera conditions with multi view gait images with mismatched train and test conditions. The proposed learning features perform better when the best performing features are fused (DLF+LDA), even for a simple NN based classifier. This shows a significant improvement in recognition accuracies can be achieved with appropriate learning features and their fusion in real world surveillance scenarios with uncooperative camera conditions (simulated with mismatched train and test conditions).

References

1. Zheng, S.: CASIA Gait Database collected by Institute of Automation, Chinese Academy of Sciences, CASIA Gait Database, http://www.sinobiometrics.com
2. Huang, L.: Person Recognition By Feature Fusion, Dept. of Engineering Technology Metropolitan State College of Denver Denver. IEEE, USA (2011)
3. Bringer, J., Chabanne, H.: Biometric Identification Paradigm Towards Privacy and Confidentiality Protection, Biometric: Theory. In: Nichols, E.R. (ed.) Application and Issues, pp. 123–141 (2011)
4. Jain, A.K.: Next Generation Biometrics, Department of Computer Science & Engineering. Michigan State University, Department of Brain & Cognitive Engineering, Korea University (2009)
5. Yampolskiy, R.V., Govindaraja, V.: Taxonomy of Behavioral Biometrics. In: Behavioral Biometrics for Human Identification, pp. 1–43 (2010)
6. Meraoumia, A., Chitroub, S., Bouridane, A.: Fusion of Finger-Knuckle-Print and Palmprint for an Efficient Multi-biometric System of Person Recognition. In: IEEE Communications Society Subject Matter Experts for Publication in the IEEE ICC (2011)

7. Berretti, S., Bimbo, A., Pala, P.: 3D face recognaition using isogeodesic stripes. IEEE Transaction on Pattern Analysis and Machine Intelligence 32(12) (2010)
8. Yuan, L., Mu, Z., Xu, Z.: Using Ear Biometrics for Personal Recognition. School of Information Engineering, Univ. of Science and Technology Beijing, Beijing 100083, yuanli64@hotmail.com
9. Ross, A., Jain, A.K.: Information fusion in biometrics. Pattern Recognition Letters 24, 2115–2125 (2003)
10. Jain, A.K., Hong, L., Kulkarni, Y.: A multimodal biometric system using fingerprints, face and speech. In: 2nd Int'l Conf. AVBPA, vol. 10, pp. 182–187 (1999)
11. Wang, Y., Tan, T., Jain, A.K.: Combining face and iris biometrics for identity verification. In: Int'l Conf. AVBPA, pp. 805–813 (2003)
12. Chang, K., et al.: Comparison and Combination of Ear and Face Images in Appearance-Based Biometrics. IEEE Trans. PAMI 25, 1160–1165 (2003)
13. Kittler, J., et al.: On combining classifiers. IEEE Trans. Pattern Anal. Mach. Intell. 20, 226–239 (1998)
14. Smith, L.I.: A tutorial on Principal Components Analysis
15. Linear discriminant analysis, Wikipedia, http://www.wikipedia.org
16. Multi Layer Perceptron, http://www.neoroph.sourceforge.net
17. Platt, J.C.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines, Microsoft Research, jplatt@microsoft.com, Technical Report MSR-TR-98-14 (17) (1998)
18. Shlizerman, I.K., Basri, R.: 3D Face Reconstruction from a Single Image Using a Single Reference Face Shape. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(2) (2011)
19. Hossain, E., Chetty, G.: Multimodal Identity Verification Based on Learning Face and Gait Cues. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part III. LNCS, vol. 7064, pp. 1–8. Springer, Heidelberg (2011)
20. Chin, Y.J., Ong, T.S., Teoh, A.B.J., Goh, M.K.O.: Multimodal Biometrics based Bit Extraction Method for Template Security, Faculty of Information Science and Technology, Multimedia University, Malaysia, School of Electrical and Electronic Engineering, Yonsei University, Seoul, Korea. IEEE (2011)
21. Multilayer Perceptron Neural Networks, The Multilayer Perceptron Neural Network Model, http://www.dtreg.com
22. UCMG Gait Database (to be made publicly available), http://www.canberra.edu.au
23. Hinton, G.E.: To recognize shapes, first learn to generate images. Progress in Brain Research 165, 535–547 (2007)
24. Huo, X., Ni, X., Smith, A.K.: A survey of manifold-based learning methods. In: Recent Advances in Datamining of Enterprise Data Algorithms and Applications, pp. 691–745 (2004)
25. LeCun, Y., Chopra, S., Hadsell, R.: A tutorial on energy-based learning. In: Predicting Structured Data. MIT Press (2006)
26. Kusakunniran, W., Zhang, J., Wu, Q., Li, H.: Multiple Views Gait Recognition using View Transformation Model of Gait Energy Image. In: Second IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS 2009): a workshop in ICCV 2009, pp. 1058–1064 (2009)
27. Chetty, G., Wagner, M.: Liveness Detection Using Cross Modal Correlations in Face-Voice Person Authentication. In: Proceedings Interspeech Conference, pp. 2181–2184 (2005)

Embedded System for Human Augmented Cognition Based on Face Selective Attention Using Eye Gaze Tracking

Bumhwi Kim, Rammohan Mallipeddi, and Minho Lee*

School of Electronics Engineering, Kyungpook National University,
1370 Sankyuk-Dong, Puk-Gu, Taegu 702-701, South Korea
bhkim@ee.knu.ac.kr, {mallipeddi.ram,mholee}@gmail.com

Abstract. This paper proposes a new embedded system which can selectively detect human face based on eye gaze information and can also incrementally recognize the human subject. The proposed embedded system comprises of one glass-type platform, two embedded modules and one android platform. The glass-type platform can detect user's eye gaze through eye camera, human faces through frontal camera and selects the face of the preferred human subject based on their gaze information. Android platform performs face recognition and information displaying operation. All modules in the system are connected and communicate wirelessly. The experimental results indicate that the proposed system is reasonable.

Keywords: bottom-up selective attention, gaze tracking, embedded platform, android platform, modified census transform, adaboost face detection.

1 Introduction

The human visual system seems to effortlessly focus and detect an area of interest to identify an object within natural or cluttered scenes with noisy environment. However, this effective task of processing complex visual scenes is performed by a cognitive function known as selective attention.

In last few years, there have been several studies to understand the process of selective attention and to model it for improvised computer vision. For instance, Itti, Koch, and Niebur [1] introduced a brain-like model to generate a saliency map (SM). SM is a brain-like vision model that imitates receptive parts of human visual cortex. It uses bottom-up features such as color, intensity, and edge information to infer salient regions and compare them with surrounding environment. Jeong and his colleagues [2] introduced a dynamic saliency model, which considers temporal dynamics of saliency changing through time at each salient point. This model is based on a modified static saliency model, which additionally considers symmetry information. Further, Ban and his colleagues [3] proposed an effective method to generate saliency map considering psychological distance as well as visual features.

* Corresponding author.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 729–736, 2013.

In cognitive psychology, it is widely assumed that human eye gaze contains important information related to human intention during a visual search task. The notions that direct gaze facilitates processing of an observed face is supported by several behavioral studies such as [4]. This study showed improved performance in gender categorization and face recognition tasks when the face stimuli display direct rather than averted gaze.

Face detection is still an active research area in machine vision. The Haar-like feature based adaboost algorithm proposed by Viola & Jones [5] and the MCT feature based adaboost algorithm proposed by Fröba & Ernst [6] are widely used algorithms among several other face detection algorithms. Haar-like feature based face detector is weak at handling illumination because Haar-like features are calculated by subtracting the summation of pixels in two different regions. On the other hand, MCT feature based face detector is robust to illumination. Therefore, in this paper, we use MCT feature based adaboost algorithm for face detection.

In human social interaction, human identification in addition to human face detection is essential for an efficient communication. But human subjects tend to forget the information which is not used frequently. Therefore, the purpose of the proposed embedded face recognition model is to support human subjects by augmenting thier memory.

Our previous research introduced an embedded system that incorporates bottom-up selective attention model into adaboost based face detection. [7] In this paper, we propose an embedded platform for incremental face recognition. The system consists of following: 1) a glass-type platform to acquire user's eye image and out-scene image; 2) two embedded modules to detect multiple faces and eye gaze to select a face based on eye gaze; 3) android platform for recognizing detected face and displaying its information.

This paper is organized as follows: Section 2 describes the proposed embedded system for human augmented cognition. The experimental results are presented in Section 3. Finally, section 4 presents conclusions and future works.

2 Proposed Embedded System

Figure 1 outlines an overall architecture of the proposed model. The input images are obtained from both frontal camera and eye camera and they are attached to the main and sub embedded processors respectively. The obtained images are transferred to the respective embedded module. The sub embedded module detects eye pupil of the user and its center position in the image. The detected position is then transferred to main embedded module which in turn detects multiple human faces based on face selective attention model and maps eye gaze position to out-scene image. Then a face is selected from the detected faces by considering eye gaze point and is transferred to an android platform. Finally, the android platform recognizes the face based on incremental two-dimensional two-directional principal component analysis ($I(2D)^2PCA$) [8]. The recognized results are displayed on android platform. The communication between the two embedded modules and the android platform happens wirelessly.

2.1 Face Selective Attention Model

In human visual processing, features such as intensity, edge and color are extracted in the retina. These extracted features are transmitted to the visual cortex through the lateral geniculate nucleus (LGN). While transmitting the features to the visual cortex, intensity, edge, and color feature maps are constructed using the on-set and off-surround mechanism of the LGN and the visual cortex. And the feature maps make a bottom-up SM model in the lateral intra-parietal cortex (LIP) [9].

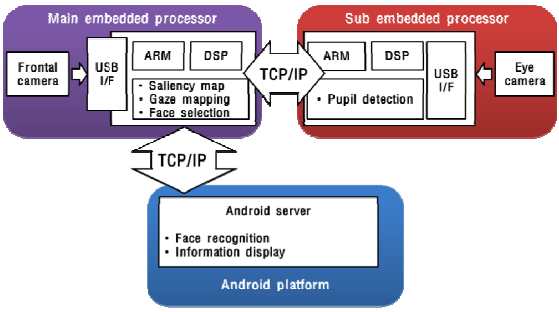


Fig. 1. The proposed system

In this paper, we used face selective attention model to detect human faces [10, 11].

The face selective attention model consists of bottom-up selective attention model and MCT feature based adaboost face detector. The bottom-up selective attention model reflects the functions of the retina cells, the LGN and the visual cortex. Since the retinal cells can extract edge and intensity information as well as color opponency, we use these factors as the basic features of the bottom-up selective attention model [11]. The MCT feature based adaboost face detector is used for correctly localizing faces in the face candidate regions by the bottom-up selective attention model. The MCT feature is a simple vector with 0 or 1 values and is robust to various illumination changes. A 3x3 moving window in an image generates the MCT features. The MCT features represent the relativeness between global average intensity and the intensity values in current window of an image [8].

Figure 2 represents the whole process of the face selective attention model.

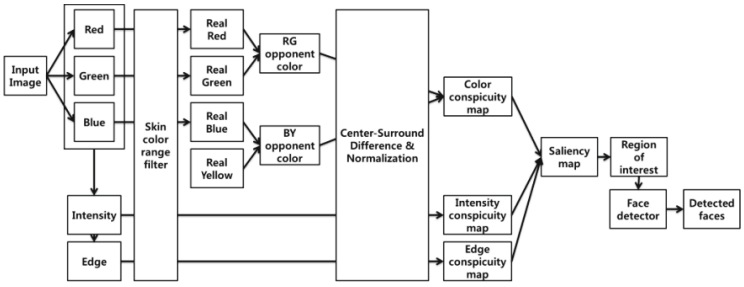


Fig. 2. Diagram of face selective attention model

From the input image, color and intensity features are extracted. Edge feature is obtained by applying Sobel operator. The filtered color features are converted to real color features. And opponent color feature is obtained from real color features. Then conspicuity maps of color, intensity and edge is obtained by applying center-surround difference & normalization algorithm. The saliency map is calculated by adding pixel-by-pixel of the three conspicuity maps. From the saliency map, region of interest (ROI) is determined. Finally, face detector operates within ROIs and detected faces.

2.2 Eye Pupil Detection

Based on psychological studies, it is evident that the human eye gaze contains significant information related to human intention. During a visual search task the human implicit intent can be classified as navigational and informational intents [12]. According the researchers in [12], there exists a significant difference in the eye movements during the two different intent conditions. Therefore, human implicit intent can be identified by analyzing human eye movements.

To obtain human eye image, we used glass-type platform which can obtain out-scene image and human eye image. Figure 3 shows the glass-type platform.

Fig. 3. The glass-type platform

2.3 Face Recognition

Forgetting is a natural human phenomenon. Ebbinghaus [13] extrapolated the hypothesis of exponential nature of forgetting. According to the Ebbinghaus's hypothesis, humans can forget recognized faces when they do not have appropriate or significant relationship with each other. Considering this limitation of human memory, this paper proposes an embedded system, which can enhance the capacity of human memory during face recognition. In face recognition, there are two types of learning models; one is batch learning and the other is incremental learning. Batch learning requires much computational time and memory usage as they collect the entire training data to compute. In contrast, incremental learning model reduces the computational time and requires less memory to store entire features. Since, the proposed model is implemented on an embedded platform, we consider the incremental learning model; $I(2D)^2PCA$. Figure 4 shows the procedure of face recognition model.

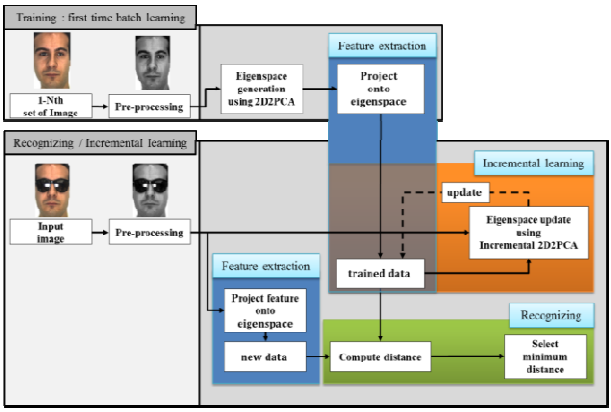


Fig. 4. Flowchart of face recognition model

Face recognition model is divided into three parts: batch training, incremental learning and recognition process. Initially, the face recognition model performs batch learning with *N* basic images. When an image is presented as an input to the system, face recognition model tries to recognize and classify the face based on previous trained dataset. If the input image belongs to the trained class, then the model gives a specific class label. If the input image does not belong to any of the trained class then the face recognition model learns the face incrementally. Compared to the batch type learning model, our face recognition model can reduce the computational time and memory requirements.

2.4 Hardware Implementation

2.4.1 Hardware System

Two embedded modules can handle complex operations such as convolution or floating point calculations in real-environment. In this paper we consider the second version (e.g. since first version [7]) towards the bigger goal of human augmented cognition in our proposed system.

The image data from the internal and external cameras are given as input to embedded modules. Then each embedded module performs the face detection and selection with the proposed model. Then detected face image is transferred to android platform. Finally, android platform recognizes the face and displays the results. Figure 5 shows embedded modules and android platform.

Fig. 5. Embedded modules and android platform

3 Experimental Results

To improve the performance of the algorithm in two embedded modules, the embedded module utilizes co-processing between DSP and ARM core. The adaboost based face detector processes a moving window operation to detect faces of various sizes. This process burdens the ARM processor with high computational cost. Therefore, we consider the DSP processor for parallel processing. Moreover, we applied detected face tracking for optimization. Once a face is detected, it is stored as a template. Image's geometrical information such as size and position is also temporarily stored. The proposed model tracks pre-stored face template along with its pre-detected geometrical information in a frame. The model can track a face even if it is not detected in current frame. Finally new detected or tracked face and its geometrical information are updated again. Figure 6 shows the comparison of the execution times for the previous face detection algorithm [7] running on ARM-only and when the same function is executed on the DSP, but called from the ARM (ARM+DSP; using c6runlib[14]) and the proposed face detection with tracking function in ARM+DSP.

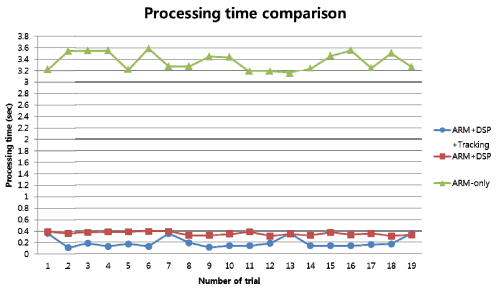


Fig. 6. The performance comparison between previous model and proposed ARM+DSP+ Tracking

From our results, it is clear that execution time is faster than previous models (faster about 6 times than previous ARM+DSP model and faster about 10 times than previous ARM model). Thus, the proposed model is proper to use in embedded system.

To test the proposed model, we also did real-world test in following outdoor environments:

- Case 1: Two people in a subway station. The user is wearing the glass-type platform, gazes the left person and recognizes it with the help of android platform.
- Case 2: A person is in a bus station. The user gazes the person and recognizes it.

In both tests, the glass-type platform with embedded module could detect and select the face. And android platform could recognize the detected face successfully. Figure 7 shows the experimental results of both tests; figure 7 shows the glass-type platform with embedded module's result in outdoor situation and figure 8 shows android platform result.

Fig. 7. Example proposed system result: selective face detection considering human eye gaze

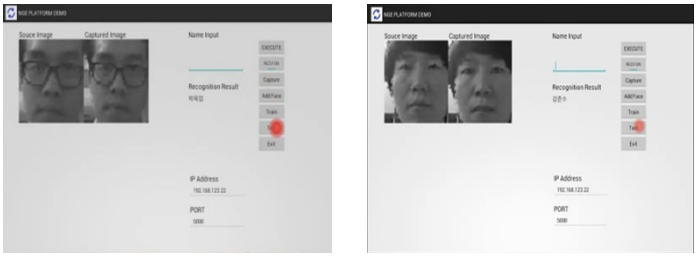


Fig. 8. Example proposed system result: android platform face recognition

4 Conclusion and Future Works

In this paper, we proposed an embedded system for human augmented cognition. The system consists of a software part (selective face detection model and incremental face recognition model) and a hardware part (glass-type platform, embedded modules and android platform). In this system, a selective face attention system based on bottom-up selective attention model with adaboost face detector using MCT feature is proposed. The face learning model based on I(2D)2PCA is used. Also the face detection with tracking architecture is proposed because of the limitations of embedded system. The experimental result shows that the proposed face detection with tracking architecture is proper to use in an embedded system.

The proposed system still has some performance issues related to design, battery and user friendly interfaces. Our final goal in this research is to make a human augmented system, which not only detects and recognizes human faces, but could also handle human voice information such as conversion between voice and text in embedded system. It should also give user relative information from recognized name, conversation and so forth. After achieving the goal, our proposed embedded system can support peoples' various cognitive issues like memory forgetting, decision making and so on.

Acknowledgement. This research was supported by the Converging Research Center Program through the Ministry of Science, ICT and Future Planning, Korea (2013K000333).

Reference

1. Itti, L., Koch, C., Neibur, E.: A model of saliency–based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(11), 1254–1259 (1998)
2. Jeong, S., Ban, S.-W., Lee, M.: Stereo saliency map considering affective factors and selective motion analysis in a dynamic environment. Neural Networks 21, 1420–1430 (2008)
3. Ban, S.-W., Jang, Y.-M., Lee, M.: Affective saliency map considering psychological distance. Neurocomputing 74(11), 1916–1925 (2011)
4. Frischen, A., Bayliss, A.P., Tipper, S.P.: Gaze Cueing of Attention. Psychol. Bull. 133(4), 694–724 (2002)
5. Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vision. 57(2), 137–154 (2004)
6. Fröba, B., Ernst, A.: Face detection with the modified census transform. In: Proceedings of the 6th IEEE International Conference on Automatic Face and Gesture Recognition (FGR 2004), pp. 91–96. IEEE Press, Seoul (2004)
7. Kim, B., Son, H.-M., Lee, Y.-J., Lee, M.: Implementation of Face Selective Attention Model on an Embedded System. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part V. LNCS, vol. 7667, pp. 188–195. Springer, Heidelberg (2012)
8. Choi, Y., Tokumoto, T., Lee, M., Ozawa, S.: Incremental two-dimensional two-directional principal component analysis (I(2D)2PCA) for face recognition. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2011), pp. 1493–1496. IEEE Press, Prague (2011)
9. Goldstein, E.B.: Sensation and perception, 4th edn. An international Thomson publishing company, USA (1996)
10. Kim, B., Ban, S.-W., Lee, M.: Improving adaBoost based face detection using face-color preferable selective attention. In: Fyfe, C., Kim, D., Lee, S.-Y., Yin, H. (eds.) IDEAL 2008. LNCS, vol. 5326, pp. 88–95. Springer, Heidelberg (2008)
11. Park, S.J., An, K.H., Lee, M.: Saliency map model with aptive masking based on independent component analysis. Neurocomputing 49, 417–422 (2002)
12. Hwang, B., Jang, Y.-M., Mallipeddi, R., Lee, M.: Probing of human implicit intent based on eye movement and pupillary analysis for augmented cognition. Int. J. Imag. Syst. Tech. 23(2), 114–126 (2013)
13. Ebbinghaus, H.: Memory: A Contribution to Experimental Psychology, Teachers College, USA (1913)
14. C6EZRun, `http://www.ti.com/tool/C6RUN-DSPARMTOOL`

Feature Selection for Stock Market Analysis

Yuqinq He, Kamaladdin Fataliyev, and Lipo Wang

School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore

Abstract. The analysis of the financial market always draws a lot of attention from investors and researchers. The trend of stock market is very complex and is influenced by various factors. Therefore to find out the most significant factors to the stock market is very important. Feature Selection is such an algorithm that can remove the redundant and irrelevant factors, and figure out the most significant subset of factors to build the analysis model. This paper analyzes a series of technical indicators used in conventional studies of the stock market and uses various feature selection algorithms, such as principal component analysis, genetic algorithms, and sequential forward search, to find out the most important indicators.

Keywords: Stock Market Analysis, Principal Component Analysis, Genetic Algorithm, Feature Selection.

1 Introduction

Stock market analysis has always been a hot area for researchers and investors. People have come up with a lot of theoretical foundation in mathematics, and developed a variety of methods to analyze the stock market with the help of modern computer technology. Among them, the Feature Selection method is a very important research field. It evaluates a lot of factors that are considered important to the stock market, and selects out the most significant ones for people to depict the market trend. The function of Feature Selection method is to discard the dross and select the essence. The computational time could be dramatically reduced, since the significant factors are pointed out for the investors. In order to figure out the prominent features in the stock market, researches on effective Feature Selection methods are extremely needed.

This paper begins with a literature review of the stock price, analysis of the stock market and feature selection algorithm. Subsequently in chapter 3, technical indicators, principal component analysis, genetic algorithm and sequential forward feature selection are given. Then, in chapter 4 the results are discussed and analyzed. Lastly, in chapter 5, we draw our conclusion and talk about future works.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 737–744, 2013.

2 Related Work

2.1 Stock Price

A stock itself has no values, but it can be set a certain price as a commodity for selling and buying, and this price is called stock price [1]. Stock price is also known as the quotation of a stock market, refers to the price of the stock traded in the securities market. The stock price is divided into two categories, theoretical-price and market-price. Theoretical-price of a stock not only provides a significant basis to predict changes in the trend of the stock market-price, but also is a basic factor for the formation of market-price in the stock market. Market-price of a stock refers to the actual price of the stock traded on the stock market. Since the stock market can be categorized into issuing market and circulation market, the market-price also has two types: the issue price and the circulation price [1]. The issue price is the price determined by the issuing company and the securities underwriter, when the stock enters the market at the first time. Therefore, when people talk about the market-price, they normally mean the circulation price of the stock. The market-price contains a lot of details, such as the opening price, closing price, highest price and many other records. The closing price is the most important one among them. It is the basic data that is used in analysis of the stock market [1].

2.2 Analysis about the Stock Market

Investors have to learn to understand a series of theories and scientific methods, in order to analyze all kinds of the information of the stock market. The most famous theory is the Efficient Market Hypothesis Theory (EMH), and the analysis method is generally divided into two types, technical analysis and fundamental analysis [2].

Efficient Market Hypothesis Theory, proposed by E.F. Fama in 1965, is a perfectly competitive market model with an entirely rational basis. It is the cornerstone of the traditional mainstream financial theory [3]. The kernel of this theory is that the stock price always tells all the relevant information accurately, adequately, and in time in an effective market.

The purpose of fundamental analysis is to determine whether the current stock price is reasonable and depict the long-term development space, whereas the technical analysis to predict the short term ups and downs of the trend of the stock price. With fundamental analysis, people can be aware of what stocks they should buy, while technical analysis helps them to detect the timing of specific purchase.

2.3 Feature Selection Algorithm

Feature Selection, also known as the feature subset selection (FSS), or attribute selection (Attribute Selection), is a method to select a feature subset from all the input features to make the constructed model better. In the practical application of machine learning, the quantity of features is normally very large, in which there may exist irrelevant features, or the features may have dependence on each other

Feature Selection can remove irrelevant or redundant features, and thus decrease the number of features to improve the accuracy of the model. The purpose of reducing the running time can also be achieved. On the other hand, selecting the really relevant features can simplify the model, and make the data generation process easy-to-understand for the researchers.

3 Theory and Methodology

3.1 Technical Indicators

Technical indicator is a proper noun in finance [4]. It refers to a collection of stock data calculated by mathematical formulas. This kind of indicators needs to take all the aspects of the market's behavior into consideration, and build a mathematical model, giving out the calculation formula, and then get a number reflecting the intrinsic essence of a certain aspect of the stock market.

There are 12 indicators in total used as the input factors for the stock market. These indicators include SMA (simple moving average), EMA (exponential moving average), ALF (Alexander's filter, which is used to estimate the percentage changes in the prices of financial varieties within a specific period), Relative Strength (it is used to compare the stock price with the whole market in a certain period), RSI (relative strength index), MFI (money flow index, which evaluates the selling and buying pressure with the help of trading price and volume), %B Indicator, Volatility, Volatility Band, CHO (Chaikin Oscillator, which measures the change of the average range of prices in a certain period), MACD (Moving Average Convergence-Divergence), %K Indicator (it focuses on the relationship between the day's high, day's low and the closing prices in the calculation process), Accumulation and distribution (AD) oscillator and Williams %R indicator (analyzes the short-term trend of the market by forecasting the high and low points in a cycle period and picking up the effective signals).

All the 12 indicators are calculated as a line vector. To form the original input feature set, all the 12 vectors are put together into a matrix called *feature*, and after the selecting process of the Feature Selection algorithms, some of them will be chosen to be the optimized feature subset. The elements of the optimized subset are exactly the most significant factors to the stock market.

3.2 Principle Component Analysis

Principle Component Analysis (PCA) is a statistical analysis method to extract the principle contradiction of things, proposed by K. Pearson in 1901 [5]. The essence of this method is to reveal the nature of things by resolving the main factors and simplifying complex problems. PCA is mainly used for dimensionality reduction of data.

The calculation purpose of PCA is to make a projection from the main components of high-dimensional data, onto a lower dimensional space. A multidimensional vector is composed of a series of examples of the characteristics, and some of the elements

have no distinction themselves. The goal is to find elements with huge changes, i.e. the dimensions with high variances, and removing those with little changes, in order to reduce the computation time.

The detailed steps of doing PCA are:

- Calculate the covariance matrix S of the sample matrix X.
- Obtain the eigenvalues (1, 2, …, N) and eigenvectors (e_1, e_2,…, e_N) of the covariance matrix X. Sort the eigenvalues in descending order.
- Project the sample data onto the space formed by the eigenvectors, and get the new sample matrix.

3.3 Genetic Algorithm

Genetic Algorithm (GA) is a kind of random search methods, which is inspired by the laws of evolution in the biosphere (the survival of fittest) [6]. This algorithm has internal implicit-parallelism and better global optimization capability. GA can automatically access and guide the optimized search space, with the probabilistic optimization methods, and there is no need of certain search rules, GA can adjust the search direction self-adaptively. These properties of genetic algorithm have been widely used in combinatorial optimization, machine learning, signal processing, adaptive control and artificial life. It is one of the key technologies in the area of modern intelligent computation. The process of the genetic algorithm includes Initialization of the Population, Individual Evaluation, Selection, Crossover and Mutation steps.

3.4 Sequential Forward Feature Selection

Sequential Forward Selection (SFS) algorithm is one kind of Heuristic searching methods. This method starts with an empty feature subset X, and adds a feature x to make the Criterion function J (X) optimal at each decision step [7]. Simply speaking, it selects a feature that could give the optimal value of the evaluation function each time, and in fact, it is a simple greedy algorithm.

However, there is a disadvantage that it could result in nesting problem, since once a selected feature is added to the subnet X, it would not be discarded any more. For example, if one feature A is completely dependent on the other two features B and C, then it is obvious that A is superfluous since B and C are the dominant factors. Suppose that the Sequential Forward Selection algorithm select feature A first, and then adds B and C, therefore the selected subset will be redundant since it could not remove feature A.

Another sequential Feature Selection algorithm called Sequential Backward Feature Selection (SBS) method has the opposite mechanism. It starts with the full set X = Y, and each time removes one feature so that the evaluation function achieves optimal. Similarly, this algorithm also has the drawback of nesting problem. It cannot add back the removed feature, and needs more computation than SFS. Therefore, the Sequential Forward Selection is used to do the selection in this part.

The selecting process of SFS is similar to a searching process. The algorithm starts with an empty set, and each time chooses one feature to make the evaluation function optimal until all the features are added into the subset or the evaluation function could not be better any more. The detailed procedures of doing the selection are illustrated below:

- Define an empty feature subset S.
- Evaluate each feature in the input feature set and choose one feature that could make the evaluation function have the optimal value.
- Add this feature into the subset S, and choose the second feature out of all the other left features with the same evaluation criterion as the previous step.
- Repeatedly evaluate the left features and add them into S until there is no improvement when adding a new feature or the criterion is met.

4 Discussion and Analysis

All the results obtained from the three algorithms are shown below together with the result got in [8]. This proposed data was evaluated by Sui et al. using Genetic Algorithm with the criterion of measuring classification complexity. From Table 1, it could be concluded that the results of the four programs turned to be good.

The result got from PCA should be relatively "correct", since the mathematics is designed to find out the most efficient dimensions to describe the target. The order of the output has specific meaning, since it is ordered according to each factor's contribution rate. From Table 1, it can be seen that the most significant feature is Volatility, and the least ones are Chaikin Osillator and Williams %R. This result is the same as the one that proposed in [8], it selects 10 features out of 12, excluding Chaikin Osillator and Williams %R as well.

In the meanwhile, the Genetic Algorithm has some randomness, which will influence the result if the iteration times are not enough. The number of features could not be controlled in the first version since there might be crossover and mutation happening at any time. Besides, the result might be different due to the randomness of Genetic Algorithm, however, if the number of generation and the size of population are large enough, the result should go into a steady situation and have optimal solution. Although this program only selects five features to form the output subset, the chosen ones are the first five features in the output of PCA. The accumulative contribution of the first five features is 94% (in Part 3.2.5), which is much higher than the normal criterion 70%. Therefore, the result of GA_1 can be concluded trustworthy and concise.

According to the explanation before, the output of GA_2 only denotes which features are selected by the subset, and there is no difference in the importance. The number of features to be selected to form the feature subset can be controlled with a parameter feaN, and the output with feaN = 10 has the same ten features as that of [8], meaning that this program also works well.

Table 1. Comparison of the results

	Technical Indicator	Proposed Result	PCA	GA_1	GA_2	SFS
1	ALF	(1)ALF	(6)V	(1)ALF	(7)VB	(4)MFI
2	RS	(4)MFI	(5)B	(3)RSI	(3)RSI	(6)V
3	RSI	(10)%K	(1)ALF	(5)%B	(11)ADO	(9)MACD
4	MFI	(5)%B	(3)RSI	(6)V	(5)%B	(7)VB
5	%B	(7)VB	(9)MACD	(9)MACD	(2)RS	(1)ALF
6	V	(6)V	(10)%K		(9)MACD	(8)CHO
7	VB	(3)RSI	(7)VB		(4)MFI	(3)RSI
8	CHO	(9)MACD	(2)RS		(10)%K	(5)%B
9	MACD	(2)RS	(4)MFI		(6)V	(2)RS
10	%K	(11)ADO	(11)ADO		(1)ALF	(10)%K
11	ADO		(8)CHO			(12)%R
12	%R		(12)%R			(11)ADO

The output of SFS has the discarded feature "Chaikin Osillator" on the sixth position, which makes the subset seem not good as those of the others. This is due to the drawback "nesting problem" of SFS which is explained in Part 3.4.1. If the feature has been selected, it could not be removed any more. Therefore, SFS might not be very efficient during actual practice.

Overall, PCA is the most suitable method in this paper, since it is reliable and accurate. However, the computation time might be very long if the input data has too many factors. In such a situation Genetic Algorithm will have a better performance since it takes the advantage of randomness.

5 Conclusion

In this paper, we did researches on the principles and theories in the field of financial market, and basic technical analysis methodologies about the stock market was studied and practiced with the help of Feature Selection algorithms. We used the data of Shanghai Stock Exchange Composite Index (SSECI) from 24/03/1997 to 23/08/2006 to measure twelve technical indicators for later research. The twelve chosen technical indicators were calculated, and the results were taken as the input of the Feature Selection algorithms. The three kinds of Feature Selection algorithms, Principle Component Analysis (PCA), Genetic Algorithm (GA) and Sequential Forward Selection (SFS) were studied. According to the results and analysis, PCA is the most reliable, but might be time-consuming if the input has very large dimensions. Genetic Algorithm will have a better performance since it takes the advantage of randomness in such a situation. SFS could generate the local optimal solution, but with a risk of "nesting problem".

Based on this paper, there are still many improvements that can be made to get better results. Here are some recommendations we summarized for a better work.

- This paper only studies on three kinds of Feature Selection algorithms, so other algorithms [13]-[19] can be researched and compared with these three to provide more convenient and reliable method for building models. Moreover, the study on a hybrid algorithm will also have great potential to come out a better solution.
- Pattern recognition techniques like Artificial Neural Network and Support Vector Machine could be used to do further analysis. By training the learning machine with the selected features, the resultant model will be more reliable and practical.

References

[1] Wise, B.O.: Fundamentals of the stock market. Machinery Industry Press (2012)
[2] Roberts, H.V.: Stock-Market "Patterns" And Financial Analysis: Methodological Suggestions. The Journal of Finance (1959)
[3] Fama, E.F.: Random walks in stock market prices. Financial Analysis Journal 21, 55–59 (1965)
[4] Blume, L., Easley, D., O'hara, M.: Market Statistics and Technical Analysis: The Role of Volume. The Journal of Finance (1994)
[5] Jolliffe, I.T.: Principle Component Analysis. Springer, New York (1986)
[6] Huang, C.L., Wang, C.J.: A GA-based feature selection and parameters optimiza-tion for support vector machines. Expert Systems with Applications 31, 231–240 (2006)
[7] Liu, B., Wang, Y., Lu, L., Li, Y., Cai, Y.: Prediction the protein SUMO modification sites based on Properties Sequential Forward Selection. Biochemical and Biophysical Research Communications (2007)
[8] Sui, X.S., Qi, Z.Y., Yu, D.R., Hu, Q.H., Zhao, H.: A novel feature selection approach using classification complexity for SVM market trend prediction
[9] Wang, L.P.: Wavelet neural networks for stock trading and prediction. In: Invited Oral presentation, SPIE Defense, Security, and Sensing, Bal-timore, USA, 29 April - 3 May (2013)
[10] Gupta, S., Wang, L.P.: Stock Forecasting with Feedforward Neural Networks and Gradual Data Sub-Sampling. Australian Journal of Intelligent Information Processing Systems 11, 14–17 (2010)
[11] Zhu, M., Wang, L.P.: Intelligent trading using support vector regression and multilayer perceptrons optimized with genetic algorithms. In: 2010 International Joint Conference on Neural Networks, IJCNN 2010 (2010)
[12] Gupta, S., Wang, L.P.: Neural networks and wavelet de-noising for stock trading and prediction. In: Pedrycz, W., Chen, S.M. (eds.) Time Series Analysis (2012)
[13] Wang, L.P., Zhou, N., Chu, F.: A general wrapper approach to selection of class-dependent features. IEEE Transactions on Neural Networks 19(7), 1267–1278 (2008)
[14] Zhou, N., Wang, L.P.: Effective selection of informative SNPs and classi-fication on the HapMap genotype data. BMC Bioinformatics 8, 484 (2007)
[15] Zhou, N., Wang, L.P.: A modified T-test feature selection method and its application on the Hapmap genotype. Genomics, Proteomics & Bioinformatics 5, 242–249 (2007)

[16] Wang, L.P., Chu, F., Xie, W.: Accurate cancer classification using expres-sions of very few genes. IEEE-ACM Transactions on Computational Biology and Bioinformatics 4(1), 40–53 (2007)
[17] Liu, B., Wan, C.R., Wang, L.P.: An efficient semi-unsupervised gene selection method via spectral biclustering. IEEE Transactions on Nano-Bioscience 5(2), 110–114 (2006)
[18] Chu, F., Wang, L.P.: Applications of support vector machines to cancer classification with microarray data. International Journal of Neural Systems 15(6), 475–484 (2005)
[19] Fu, X.J., Wang, L.P.: Data dimensionality reduction with application to simplifying RBF network structure and improving classification performance. IEEE Trans. System, Man, Cybern, Part B-Cybernetics 33(3), 399–409 (2003)

Hierarchical Classification of Vehicle Images Using NN with Conditional Adaptive Distance

Fabrízia Medeiros de S. Matos[1,2] and Renata Maria Cardoso R. de Souza[2]

[1] Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, João Pessoa, Brazil
fabrizia@ifpb.edu.br
[2] Centro de Informática, Universidade Federal de Pernambuco, Recife, Brazil
rmcrs@cin.ufpe.br

Abstract. A vehicle classification is essential for effective transportation system, parking optimization and law enforcement. Proposed methods of vehicle image classification, obtained from videos of road traffic, have known limitations, such as dependence on detection methods, hard image normalization and low accuracy. This paper presents a classification method for vehicle images using NN with conditional adaptive distance. Its aims are to improve the classification accuracy, evaluate the conditional adaptive-NN rule and to discuss the feasibility of the features based on the edge. Results are compared with NN, adaptive-NN and SVM. The experimental platform is built on Matlab R2009a.

Keywords: traffic parameters, vehicle image classification, adaptive distance.

1 Introduction

Traffic parameters are required in an effective transportation system, such as vehicle count, speed, congestion level, movement of vehicles in the intersection, vehicle classification, vehicle identity via the number plate, identification of suspected behavior, passenger count and number of accidents. Various kinds of traffic control system, for example, law enforcement, automatic tolls, automatic routing in congestion and incident detection can be implemented with these parameters [1].

Automatic vehicle classification is an important task in Intelligent Transportation System (ITS) because it enables the attainment of traffic parameter called vehicle count by category, which is used to control and manage the roadway traffic. Vehicle class is noteworthy both for statistic purposes and access control, as closed areas for some vehicle classes and limits of velocity [2].

The application of image processing and computer vision techniques to the analysis of video sequence of traffic flow offers considerable improvements over the existent methods of traffic data collection and road traffic monitoring. Challenges are not so simple: a vision-based system for such traffic applications must have the features of a short processing time, low processing cost and high reliability. As the images are represented in the 2D space, some inaccuracies occur in the acquisition

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 745–752, 2013.

system, because the real context corresponds to 3D scenes. Moreover, noise is expected, mainly, due to vehicle movement and climatic variations [3].

Real scenes consist in overcoming challenges to methods for collecting traffic parameters based on image processing. Lighting and climatic variations, shadows away from the camera, image quality, dynamic positioning of vehicles on the roads and heavy reliance on detection techniques are considered aspects that hinder the collection automation of traffic parameters from videos, by using image processing. Errors in the detection and classification of vehicles from video may be significant, particularly when traffic congestion is high, because the occlusion is inevitable [4].

The vehicle image classification proposed in this paper is based on vehicle edge images and NN with conditional adaptive distance. Detection of vehicles in the video and normalization are not covered by the proposed method.

Four labels (motorcycle, car, bus and truck) are assigned by classification. The four classes are defined based on reports of the Brazilian Ministry of Transport [5]. The separation of the bus and truck classes, instead of considering only large vehicles, is important for this ministry because there are planning, investments and political decisions, related to public transport, focused on vehicles of the bus class.

This paper is organized as follows: related works are presented in Section 2; details of the proposed classification methods, features and test methodology are in Section 3; reports of the experimental results are presented in Section 4. Finally, a conclusion is presented in Section 5.

2 Related Works

Neural-edge-based for vehicle detection and classification was presented in [6]. The height and width of a rectangle region, detected as a vehicle, were placed in input lines of a feed-forward network that does the classification into three categories: small, medium and large. The network was trained with back propagation and 3000 samples collected under weather and lighting conditions. Vehicle classification rate reached above 95%.

Morphological techniques are used for vehicle image classification in [7]. After a second level concerning morphological dilatation and removed unwanted objects, the image is binary and ready for classification. The total number of pixels in each vehicle is calculated and based on this feature, a vehicle is classified into small (100-400 pixels), medium (401-700 pixels) and large (701 - 1000 pixels), by fixing some measurement range values.

By using an analysis of time-spatial image (TSI), seven features (width, area, compactness, length-width ratio, major axis, minor rectangularity and solidity) are obtained from segmented image for vehicle classification [4]. Five classes are considered and error rate ranges from 6,9% up to 14,29%, depending on the controlled time conditions (normal, summer and clouds).

Edge-based features, as number of edge points, block of PCA and two classification levels were proposed in [8]. The classification level 1, based on the number of edge points and on the distance to prototype, divides the vehicles into two

groups, small and large. Afterwards, using PCA blocks of the original image as features and applying NN and A-NN[9], the classification level 2 divides each group into two classes: the small group is divided into motorcycle and car; the large group is divided into bus and truck. Twenty images were used in the tests, five in each class. The accuracy was 100% and 95%, respectively, for classification level 1 and level 2.

A method that trains SVM with MBF and HOG is proposed in [10]. A background Gaussian Mixture Model (GMM) and shadow removal method have been used to deal with sudden illumination. Majority vote over five consecutive frames makes the final classification. Classification accuracy of 94% was achieved.

2.1 NN with Adaptive Distance (A-NN)

NN with adaptive distance is a classification algorithm that penalizes the distance between classification object and training examples positioned close to decision boundary.

Comparisons of error rates associated to the use of NN, A-NN and SVM classifier, considering five databases from the UCI machine learning repository and Euclidean and Manhattan distances were presented in [9]. In all considered databases, results based on A-NN overcame NN rule.

3 Proposed Method

Proposed method for vehicles image classification is a classification process including pre-processing, training and classification steps. It classified a test image into one of the four classes: motorbike, car, bus and truck.

3.1 Pre-processing

The pre-processing step carries out the following processing phases: background subtraction, noise reduction, edge detection and non-contiguous elimination of horizontal and vertical bands. These five pre-processing steps were proposed in [11].

3.2 Training

The training step makes the following feature extractions: edge point number (NP), width (W), height (H), fractal dimension (FD), average of block width (BW) and average of block height (BH). The first four features (NP, W, H, FD) are described in [11] and the features BW and BH are described in [12].

Firstly, each feature values are calculated for all database vehicles. Each feature is, then, normalized by fulfilling its division (of the feature value) by the maximum value of the respective feature, considering in the maximum value calculation all the vehicles of the database.

Afterwards, the conditional radius and the neighbor conditional radius are calculated. These radiuses, which are proposed in this paper, are separately calculated

by each feature of each training example. With these conditional radius is made a simplification in the A-NN rule, considering that the distance will be penalized by the radius only when the extern radius is greater than the intern radius.

Conditional Radius (cr): Being *i* a training example of *c* class. External radius (*er*) of *i* is calculated according to [9]. In a similar way, the internal radius (*ir*) of *i* is calculated, considering only samples of the *c* class. The distinction with reference to calculus of *er* and *ir* consists only in the set of training examples such as: calculating *er* examples which do not belong to the *c* class are considered whereas in *ir* calculation only the examples of the *c* class are considered.

The conditional radius of the example *i* is defined according to:

```
if er ≤ ir
  cr = er
else
  cr = 1
end
```

Neighbor Conditional Radius (ncr): Let us assume that $f_{c,i}$ is the value of the feature *f* of the example *i,* that is of class *c*, that there are *a* classes and *b* examples in each class. The *ncr* of *i*, associated to the *f* feature, is calculated in accordance with the following pseudo-code:

```
1. for x=1 to a
    1.1 for y=1 to b
         1.1.1 d1x,y = |fc,i - fx,y|
3. for x=1 to a
    3.1 d2 = d1x,1:b
    3.2 the k lowest vales of d2 and its number of
        ocurrences are stored, respectively, in d3 and n
    3.3 px = ∑k(d3*n)
4. p_int = px ; to x = c
5. p_ext = min(px) ; to x ≠ c
6.if p_ext <= p_int
    ncri = p_ext
   else
    ncri = 1
```

3.3 Classification

The used classification is a hierarchical process based on [8]. Two classified levels are defined: level 1 classifies a test image into small or large group; and level 2 categorizes each group into its respective classes. For example, if level 1 sets the label small group to a test image, in level 2, the test image will only be classified as motorcycle or car.

Two conditions for widening the adaptive distance are defined: the first one is based on the conditional radius (*cr*); and the second one is based on the neighbor conditional radius (*ncr*).

Let us assume that there are n training examples, f features and i is a training example, with $i = [1, ..., n]$. Being X_i the feature value vector of i, R_cr_i the radius *cr* value vector of i and Y the feature value vector of the test image. The classification rule for conditional A-NN, based on *cr*, is presented in the following pseudo-code. The neighbor conditional A-NN is obtained only by exchanging *cr* by *ncr*.

```
for t=1 to n
  d_t = |X_t - Y|
  cond_d_t = ∑ (d_t / R_cr_t)
end
class = label of min(cond_d)
```

Classification levels 1 and 2 use conditional adaptive-NN distance. In level 1, the radiuses are calculated by considering two groups, small and large. In this level, all the vehicles of the motorcycle and car class are considered as small group, and all the vehicles of the truck and bus class are referred as large group. In level 2, the radiuses are also calculated by considering two classes. Nevertheless, these two groups correspond to the two internal classes of the respective group.

In defining external class of a training example, in level 1 one group is defined and in level 2 a class is characterized. For example, if a training example is a car, so for calculating the radiuses of levels 1 and 2, the external class corresponds, respectively, to the large group (truck and bus) and the motorcycle class. Similarly, if the training example is a truck, so for calculating the radiuses of levels 1 and 2, the external class corresponds, respectively, to the small group (motorcycle and car) and the bus class.

3.4 Test Methodology

The used dataset image is the same dataset used in [11]. In Figure 1, one image of each of the four classes were presented in sequence (motorcycle, car, bus and truck). There is neither centralization, nor illumination normalization.

Fig. 1. Vehicle image examples (motorcycle, car, bus and truck)

By leaving one image out, it was meant to divide the dataset into training and test. Being all dataset $B = \{I_{c,1}, I_{c,2}, ..., I_{c,m}\}$, $c \in [1,4]$ and $m = 25$. When the test image is $I_{p,q}$ four images are removed of the dataset (one of each class) and the training set becomes $B' = B - \{I_{1,q}, I_{2,q}, I_{3,q}, I_{4,q}\}$. Four images are removed in order to maintain equivalent the number of representatives from each class.

Every combination of six features (from 1 to 6) is made, and each one is used as input to four classifiers: NN, A-NN, conditional A-NN and SVM (kernel linear).

The experimental platform is built on Matlab R2009a.

4 Results and Discussions

Figure 2 presents the results of classification by considering six features, used alone, and 4 classification rules. The features in sequence are: NP, W, H, FD, BW and BH. The classification rules applied here are without level, that is, a test image is compared with all training examples and the label is related to one of the four classes (motorcycle, car, truck or bus). Conditional A-NN reaches better results with five of the six features analyzed, and the use of the *cr* radius presents better results than the *ncr* radius. Considering the average results, A-NN, Conditional A-NN with *cr* and Conditional A-NN with *ncr*, reach, respectively, 39.43 and 43.83 and 37.5.

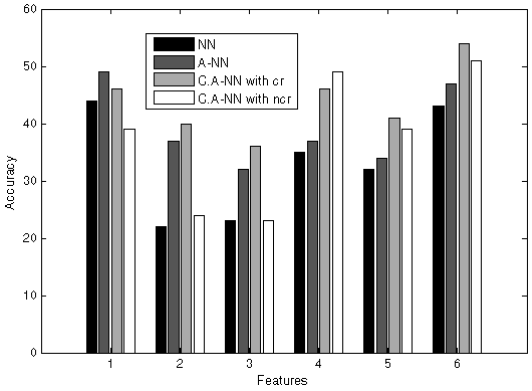


Fig. 2. Accuracy of six features versus four classifiers

Accuracy and number of errors of the better results of level 1, of each classifiers, is presented in Table I. These better results were all reach with two features.

Table 1. Level 1: Accuracy and number of classification errors, per class

Classifier	Features	Accuracy (%)	Number of errors			
			Motorcycle	Car	Truck	Bus
A-NN	PN + FD	91	2	4	1	2
C.A-NN, cr	PN + BH	92	1	4	1	2
C.A-NN, ncr	PN + BH	92	1	4	1	2
SVM	PN + H	86	2	9	1	2

The errors in level 1 classification are heavily concentrated in the small group, of the car type. These errors occur because many car images have more than one car; consequently the width and height of these images are not suitable.

Figure 3 presents the classification results of level 2. Results of level 2 represent final classification accuracy for 100 tests and the number of tests of level 2 depends on the right classification of level 1. All combinations 2 x 2 of the six features were used. The A-NN, conditional A-NN (with radius *cr* and *ncr*) and SVM classifiers are presented. In the best result (69%), reach with conditional A-NN with *cr*, the errors are concentrated in separation of the truck and bus classes.

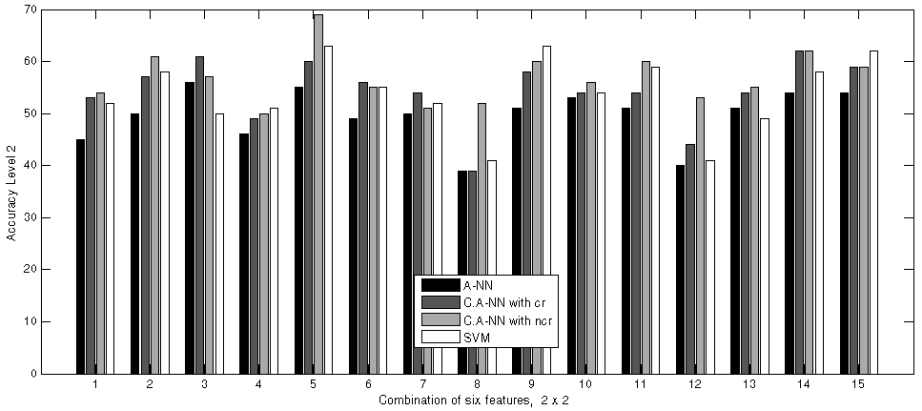


Fig. 3. Level 2 accuracy: 15 set of features versus 4 classifiers

Considering this database, in [11] achieved accuracy was 55%. With hierarchical classification and conditional A-NN with *cr*, the proposed method reaches 69%.

5 Conclusion

In the test, the results of conditional A-NN are slightly better than A-NN. This suggests that the proposed classification rule makes it an appealing tool for pattern recognition. In this database, using only one feature, the errors of A-NN occur because some radius values are equal to zero, and this produces draws.

The final accuracy (69%) is low for real applications of traffic. So, the method need to be improved because the error tolerated in this area is approximately 5%.

If there is more than a vehicle in the image, features based on the vehicle edge which are used, in an isolated or combined form, do not represent well the scene and, consequently, produce an accuracy rate of low classification, even when the classification rules, such as SVM, are used. Furthermore, even when images with one single vehicle are considered, edge-based features present limitations while describing the difference between a truck and a bus because these two classes demonstrate similar width and height.

Future works include analysis of new features for the classification. The conditional A-NN should also be tested in another database.

References

1. Kafai, M., Bhanu, B.: Dynamic Baysiane Networks for Vehicle Classification in Video. IEEE Transaction on Industrial Informatics 8(1), 100–109 (2012)
2. Wang, W., Shang, Y., Guo, J., Qian, Z.: Real-Time Vehicle Classification Based on Eigenface. International Conference on Consumer Electronics, Communications and Networks, pp. 4292-4295, Lai (2011)
3. Sanchez, A., Suarez, P., Conci, A., Nunes, E.O.: Video-based Distance Traffic Analysis: Application to Vehicle Tracking and Counting. Computer Science and Engineering, Scientific Image Processing 13(3), 38–45 (2012)
4. Mithun, N.C., Rashid, N.U., Rahman, S.M.M.: Detection and Classification Using Multiple Time-spatial Image. IEEE Transaction on Intelligent Transportation System 13(3), 1215–1225 (2012)
5. Brazilian ministry of transport (June 2013), `http://www.transportes.gov.br/conteudo/53887/`
6. Ha, D.M., Lee, J.M., Kim, Y.D.: Neural-edge-based Vehicle Detection and Traffic Parameter Extraction. Image and Vision Computing 22(11), 899–907 (2004)
7. Ranga, H.T.P., Kiran, M.R., Shekar, R.S., Kumar, S.K.N.: Vehicle Detection and Classification Based on Morphological Technique. In: International Conference on Signal and Image Processing (ICSIP), pp. 45–48 (2010)
8. Matos, F.M.S., Souza, R.M.C.R.: An Image Vehicle Classification Method Based on Edge and PCA Applied to Blocks. In: IEEE International Conference on Systems, Man, and Cybernetics, pp. 1688–1693 (2012)
9. Wang, N., Neskovic, P., Cooper, L.N.: Improving Nearest Neighbor with a Simple Adaptive Distance Measure. Pattern Recognition Letters 28, 207–213 (2007)
10. Chen, Z., Ellis, T., Velastin, S.A.: Vehicle Detection, Tracking and Classification in Urban Traffic. In: IEEE International Conference on Intelligent Transportation System, pp. 951–956 (2012)
11. de Matos, F.M.S., de Souza, R.M.C.R.: Vehicle Image Classification Based on Edge: Features and Distances Comparison. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part IV. LNCS, vol. 7666, pp. 691–698. Springer, Heidelberg (2012)
12. Matos, F.M.S., Souza, R.M.C.: Vehicle Image Classification Method Using Edge Dimensions, SVM and Prototype. In: International Conference on Artificial Intelligence, ICAI (2013)

An Effective Retrieval Method with Semantic Networks for Mobile Life-Log of Smartphones

Hu Xu and Sung-Bae Cho

Dept. of Computer Science, Yonsei University
134 Shinchon-dong, Seodaemoon-gu, Seoul 120-749, Korea
xuhu357@sclab.yonsei.ac.kr,
sbcho@cs.yonsei.ac.kr

Abstract. Smartphone is now playing an important role in our daily life. The various sensors and multiple features of smartphones can collect abundant information about people's life experience, which is called mobile life-log. Life-logs are useful for people to recall past meaningful events and remind of meaningful experiences. However, as the amount of life-log becomes vast, it requires an efficient way to organize and store such vast information. Semantic networks, which are used to implement human retrieval system, can help people to effectively organize and retrieve life-logs collected from smartphones. In this paper, we propose a method of mobile life-log search based on semantic relation in semantic networks. By using semantic relation, we reduce the unnecessary intermediate results so as to reduce the usage of computer memory and retrieval time. To demonstrate the usefulness of the proposed method, we constructed mobile life-log semantic networks and conducted experiment on it. As a result, the proposed method is useful for life-log retrieval in the semantic networks.

1 Introduction

Smartphones are very popular and become a personal assistant in our daily life. With the rapid popularization of smartphones and a variety of practical applications, it becomes possible to collect large amounts of personal life experience of daily life, which is called life-log. Smartphones can automatically generate and store various life-logs, which contain such information as call history, SMS (short message service), photos, music files, video files and GPS (global positioning system). As the amount of these life-log information becomes larger, it is difficult for people to effectively store and manage such huge information. One may suffer great difficulties from recalling certain events experienced before. It requires an effective way to organize, store and retrieve these personal life-logs.

Semantic Network has an advantage of storing mobile life-logs, because information is saved as interconnected network just like human long-term memory [1]. Semantic network can help people recall some specific objects, which were associated to

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 753–760, 2013.

particular events. With semantic network, people can manage information in human-centered manner.

There exists various search methods in semantic networks. A traditional search system does not return the best relevant results to users, because the results returned to user contain large amount of unnecessary intermediate results. Users should be very patient to pick up their interests from enormous returned results [2].

In this paper, we propose a search method using semantic relation. By using semantic relations and their weight values during search process, it is possible to reduce the unnecessary intermediate results so that reduce the usage of computer memory and retrieval time.

2 Related Works

Human's memory captures life in terms of events and experience. People tend to be more easily remember materials on subjects that they already know, since the information has more meaning to them and can be mentally connected to related information that is already stored in their long-term memory [3].

Human memory structure is interconnected network structure. When people pay attention to some information, related information will be activated.

Semantic Networks are designed to implement human retrieval system. It is possible to retrieve related information like the way human memory retrieval mechanism. There are several searching methods in semantic networks. Keyword-based searching method and spreading activation associative searching method are the well-known methods.

The amount of returned results is an important factor in semantic network search. Keyword based searching method is very useful one. Keyword search method enables information discovery by providing a simple interface. Even more, the user does not need to know either a query language or the structure of the network [4][5]. However, the returned results could be very huge, which is the main disadvantage of traditional keyword based searching method. Hristidis designed interactive spreading activation searching method[6]. During the search, users can select what they are interested among the relevant results. However, the relevant results could be so huge that user may have difficulties in finding what they are interested in. In order to solve this problem, Oh [7] proposed a mobile life-log search using semantic memory learning and priming. Priming based on spreading activation makes it possible to search the data with insufficient and inexact queries and relatively less returned results.Tran uses top-k explosion of query candidates mechanism to calculate the similarity of keywords[8].

In this paper, we propose a mobile life-log search method. We exploit the semantic relation during the search process. With utilizing semantic relation and calculation of its weight value, we reduce the useless intermediate results, the usage of computer memory and retrieval time. By using semantic relation, we could retrieve some nodes with low weight value, which cannot be seen during the expanding procedure.

3 Mobile Life Log Network

Mobile life-log is the information we get from smartphone devices. The mobile life-logs consist of call history, SMS, photos, music files, videos, GPS, Bluetooth, weather, emotion, and so on. These mobile logs should be structured and stored in a way that people can easily retrieve. In order to make life-logs useful, we construct mobile life-log semantic network to manage them.

Fig. 1 shows the mobile life-log network system structure. The user logs are collected from the mobile phones and stored as XML documents. After XML parsing and network generation process, mobile life-logs network is created. User inputs queries and system gives back the user with filtered results from the search process.

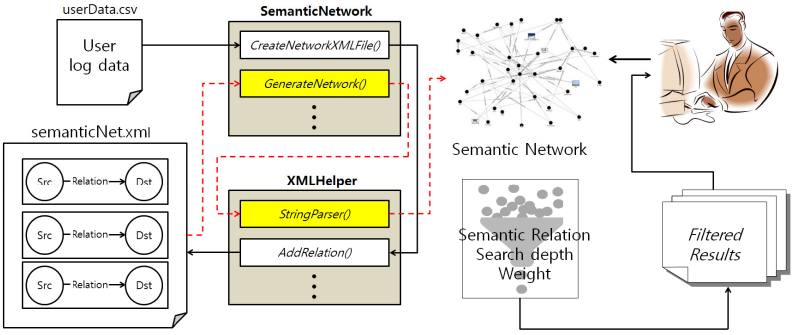


Fig. 1. Mobile life-log network system structure

3.1 Mobile Life Logging

Smartphones are very powerful tools. They equip with computing power and audio, visual sensors such as camera and microphone, as well as contextual sensors such as GPS, lighting, and RFID. With these different sensors embedded in smartphones, we can collect a variety of personal information. Other information such as weather and emotion can also be collected through smartphone applications.

3.2 Semantic Relation

Semantic networks consist of nodes and arcs. In mobile life-log semantic network, the nodes represent life-log objects and arcs represent the relationship between nodes. Every arc or edge has its own weight value.

To construct mobile life-log semantic network, we need to define semantic relation to connect related information. Activity based organization of personal life-logs could not only provide effective visualization and retrieval of information, but also benefit of augmenting human memory [9].

Semantic relation we use in this paper is defined as follows:

Definition (Semantic Relation): According to the GSS (General Social Survey on Time Use), people's activities can be categorized in several groups. In our system, we define semantic relations based on the activities which are from GSS. The main semantic relations are defined as Table 1.

Table 1. Defined semantic relation distribution

Semantic Relation	Count
General semantic relation (Is A et al.)	12
Paid work related	90
Household work	12
Sleep, meal and related work	36
Education and related activities	36
Socializing activities related	12
Passive leisure related	126
Entertainment event related	48
Active leisure related	258
Social and voluntary activities related	6
Residual	18
Total	654

3.3 Semantic Relation Weight

In semantic network, every relation has its weight value. Weight value is important factor that determines the intimacy between connected two nodes. The higher the relation weight value is, the more intimate two connected nodes are. When a node is paid attention, the neighbor nodes with higher weight value than the threshold value will be activated.

If all the related nodes activated and return to users, it is very difficult to find what users interested in. To avoid this problem and to represent the association between two nodes, we calculate the weight value of semantic relation. The calculation of semantic relation weight value is indicated as equation (1)

$$Wv,u = c_1 Nv,u \frac{\sum_{a \in Bv} Wa,v}{\sum_{b \in Fv} Nv,b} \tag{1}$$

Where Wv,u shows the weight value of semantic relation between start node v and end node u. Nv,u means the frequency of start node v and end node u appear together. Bv means the sets of nodes pointing at node v. Fv means the set of nodes pointed by start node v. c_1 is the normalized constant.

4 Proposed Method

In this paper, we propose a retrieval method using semantic relation. In mobile life-log network, when users input queries, system will find the corresponding nodes and relations.

There are keyword input box and relation selection dropdown list menu in the user interface. With keyword input box, users can find keyword matching node quickly. When user type in keyword and click search button or click the node they are interested in and semantic relations related to the node will appear in the drop down list menu.

What's more, to control the number of nodes shown to users, users can define searching depth. When one node is paid attention, neighbor nodes will be activated. However, not all the neighbor nodes are shown to users. When a semantic relation is specified, nodes that satisfying the semantic relation will be shown to users. When users do not specify any relation or system cannot find the semantic relation, nodes with higher weight value than the threshold will be shown to users no matter regardless of relations.

Semantic relation based search method has an advantage that it could avoid missing some important information despite of its low weight value. Without using semantic relation, it could not even be seen by users.

When users choose semantic relations which they are interested in, all nodes satisfying semantic relation will be returned. Fig. 2 shows the mobile life-log network of a student for one day. Among them there is a node named Funeral_Grandpa with low weight value. Funeral of grandpa is important to someone in spite of its low frequency. Without using semantic relation, the node Grandpa will not be shown.

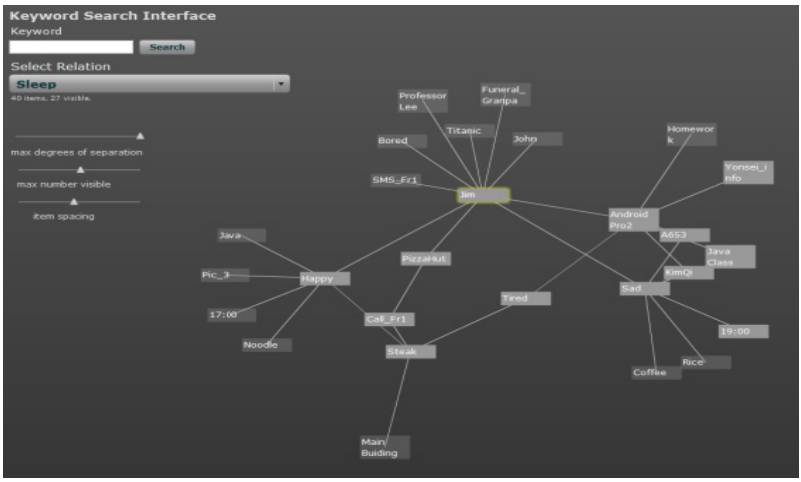


Fig. 2. Mobile life-log network of a student for one day

Fig. 3. Shows the advantage of using semantic relation to get the low weight value node. As shown in Fig. 3, the results just contain the nodes satisfying socializing activity related relation.

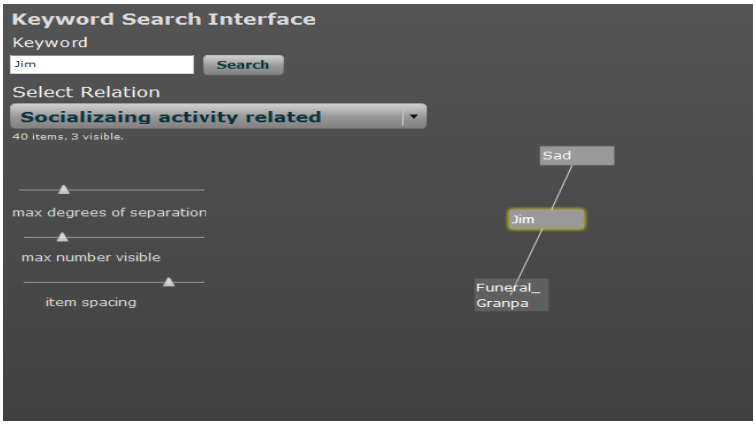


Fig. 3. After user types Jim and selects the socializing activity related relation from the relation list and click the search button. Funeral_Granpa node was appeared

The network search algorithm is as follow:

```
Input: Keyword, semantic relation
Output: nodes

Add Keyword to SortedOpenList

if semantic relation is null Then
    Extract relations related to using keyword
    if relationList is not empty, Then
        if relation.weight > threshold_weight_value
            semantic relation = relation
while SortedOpenList is not empty, Loop
        Remove the first node from SortedOpenList
    if removed node is targetNode, Then
        construct path from keyword to targetNode
        return path
    else
        Extract neighbours of the removed node
        if neighbour satisfies semantic relation, Then
            Add to Neighbours List

        while Neighbours List is not empty, Loop
            neighbour.parent = the removed node
            if Size of Path is smaller than depth_threshold, Then
                Add neighbour node to SortedOpenList
                Add the removed node to ClosedOpenList
            else
                continue loop
```

Fig. 4. Network search Algorithm

5 Experimental Result

5.1 Implementation

In order to show the usefulness and practicality of proposed method, we implement the semantic relation based mobile life-log search system and do some experiments. The system is implemented in Java and Flex.

We collected different kinds of life-logs from the people who work in different kind of jobs. We constructed 35 different kinds of scenarios. Each scenario contains one week life-logs for one person. Corresponding to these 35 scenarios, we made 62 queries to test the searching capability. The logs we use are as follows: video, picture, e-mail, schedule, GPS, call history, SMS, people, weather, activity, time and emotion logs. People, activity and emotion logs are annotated by the user and weather logs are collected from the Internet.

5.2 Evaluation

In order to evaluate the searching method, we use 62 queries to perform searching. Each query is performed 10 times. We set the searching depth from 2 to 6. As can be seen from Fig. 5, the result gets good from depth 4. When searching depth is 2 or 3, the performance is bad, because the searching operation will be stopped before arriving at the target node.

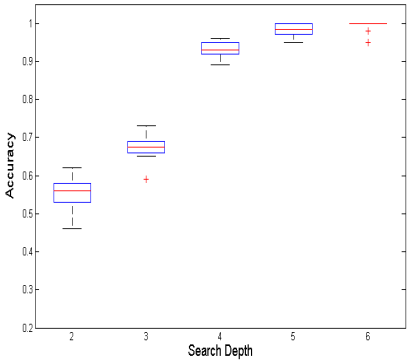


Fig. 5. Accuracy of method using relation

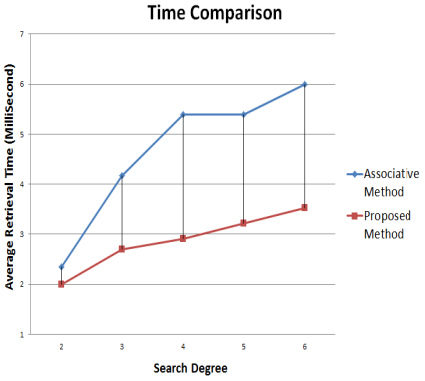


Fig. 6. Time comparison of associative method and proposed method

Fig. 6 shows the retrieval time comparison. To the retrieval time comparison experiment, we also performed searching to 62 queries setting the search depth from 2 to 6. Each one repeats 10 times. As can be seen in Fig. 6, the method using semantic relation is faster than the other. The reason is very clear that the associative method considers all the nodes at spreading activation procedure and does graph exploration much more than the semantic relation based search.

6 Conclusion

In this paper, we have proposed a mobile life-log retrieval method using semantic relation. We use the life-logs collected from smartphones to construct life-log semantic network. We calculate the weight value of relations in semantic network. By using semantic relation, we can retrieve the target information with expanding fewer nodes. Semantic relation based search can also retrieve the information with low weight value. In order to show the usefulness and practicality of the proposed method, we showed searching procedure and conducted experiments on accuracy and retrieval time of the proposed method. From the experiments, we have confirmed the usefulness of the proposed method. For the future work, we are trying to improve the searching algorithm to reduce the time complexity. Along with this, we will work on using multi semantic relations during the search procedure.

References

1. Raaijmakers, J.G., Shiffrin, R.M.: Search of associative memory. Psychological Review 88, 93–134 (1981)
2. Le, T., Vo, B., Duong, T.H.: Personalized Facets for Semantic Search using Linked Open Data with Social Networks. In: Third International Conference on Innovation in Bio-Inspired Computing and Applications, pp. 312–317 (2012)
3. Tjondronegoro, D., Chua, T.-S.: Transforming Mobile Personal Life Log into Autobiographical Multimedia eChronicles. In: Proceedings of 10th International Conference on Advances in Mobile Computing and Multimedia, pp. 57–63 (2012)
4. Agrawal, S., Chaudhuri, S., Das, G.: DBXplorer: A system for keyword-based search over relational databases. In: ICDE, pp. 5–16 (2002)
5. Guo, L., Shao, F., Botev, C., Shanmugasundaram, J.: XRANK: Ranked Keyword Search over XML Documents. In: Proceeding of the 2003 ACM SIGMO International Conference on Management of Data, pp. 16–27 (2003)
6. Hristidis, V., Balmin, Y.P.A.: Keyword proximity search on XML graphs. In: Proceeding of 19th International Conference on Data Engineering, pp. 367–378 (2003)
7. Oh, K., Yi, S.-H., Cho, S.-B.: Mobile life-log search using semantic memory learning and priming. Journal of KIISE: Software and Applications 40, 141–154 (2013)
8. Tran, T., Rudolph, S., Cimiano, P., Wang, H.: Top-k explosion of query candidates for efficient keyword search on graph-shaped(RDF) data. In: Proceeding of the 2009 IEEE International Conference on Data Engineering, pp. 405–416 (2009)
9. Tian, Y., Rao, J., Wang, W., Chen, C., Ma, J.: The Organization of Mobile Personal Lifelog by Activity. In: Proceedings of 11th Pacific Rim Conference on Multimedia, pp. 31–42 (2010)

Distance Metrics for Time-Series Data with Concentric Multi-Sphere Self Organizing Maps

Tom Gedeon, Lachlan Paget, and Dingyun Zhu

Research School of Computer Science, College of Engineering and Computer Science,
The Australian National University, Canberra, ACT 0200, Australia
{tom.gedeon,dingyun.zhu}@anu.edu.au, lachlan.paget@gmail.com

Abstract. Self-Organizing Maps have been shown to be a powerful unsupervised learning a tool in the analysis of complex high dimensional data. SOMs are capable of performing topological mapping, clustering and dimensionality reduction in order to effectively visualize and understand data and it is desirable to apply these techniques to time–series data. In this project a novel approach to time-series learning using Concentric Multi-Sphere SOMs has been expanded and generalized into a unified framework in order to thoroughly test the learning capabilities. It is found that Quantization and Topological Error are not suitable to test the learning performance of the algorithms and it is suggested that future work focus on developing new error measures and learning algorithms.

Keywords: Spherical SOM, time series, topological error, quantization error, concentric multi-sphere SOM (CSM-SOM).

1 Introduction

Self-Organizing Maps (SOMs) are primarily used for clustering, classification, sampling and visualizing high dimensional data [1]. This technique has been widely applied in many ways, for instance clustering high-frequency financial data. The conventional neighborhood arrangements are planar SOM made of two-dimensional rectangular or hexagonal lattices. However, the planar SOM has a disadvantage which is the "border effect" [2]. During training, the neurons compete with others. The weight of the winning neuron and its neighbor are updated. Ideally, all the units have the same chance to be updated. However, in the planar map, the units at the border of the map have fewer neighbors than the inside units. At the end of training, the map may not form expected similar regions of the data space, since there are many units with unequal chances of being modified during training [3]. Therefore, many spherical SOMs were proposed in order to solve that problem.

The motivation of this research is to provide a method that users can use an arbitrary number of spheres, and to observe the results of clustering data as well as the quality of SOMs on multiple spheres. I this paper, we consider the Spherical SOM introduced in [4] as the base for our Concentric Multi-Spherical SOM (CMS-SOM) topology.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 761–768, 2013.

2 Spherical SOM (SSOM)

Traditional Kohonen Self-Organizing Maps feature a non-uniform neighborhood structure in which neurons on the periphery of the map have fewer neighbors than those in the interior. This topological asymmetry is known as the border effect and results in edge neurons suffering from a reduced probability of adaption and consequently their weight vectors "collapse into the center of the input space" [5]. A discussion of the various approaches to avoid this problem can be found in [6] which includes the solution of joining the left, right and top, bottom map edges to effectively create a toroidal map. The unfortunate consequence of toroidal maps is that they are difficult to visualize effectively in three-dimensions.

Spherical Self-Organizing Maps (SSOM) were first introduced by Ritter [7], and have the ability to eliminate the border effect [8] and can also be readily and easily visualized in 3D. SSOM, like traditional SOM have been shown to be effective tools of data analysis and visualisation in a diverse range of fields including medical [9] and financial [10]. Several approaches have been proposed for the construction of the of the spherical feature map. Sangole and Leontitsis [4] proposed a method using recursive subdivision of the Icosahedron to produce an approximately uniform distribution of neurons on a sphere. The size of the data structure used to store the adjacency information scales poorly with the sphere size (number of recursive subdivisions) but the lookup time is improved as a result [6]. Pre-processing of the adjacency matrix can be used as a work-around but this implementation still suffers from the inability to create spheres with an arbitrary number of neurons. The number of neurons N is shown below as an exponential function of the number of recursive subdivisions R of the Icosahedron: $N = 2 + 10(4^R)$.

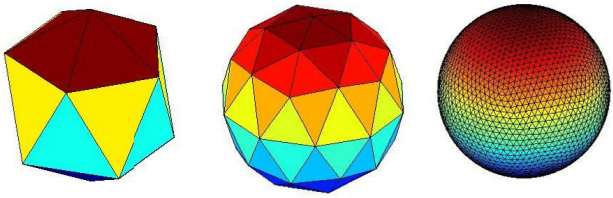


Fig. 1. Icosahedron subdivided 0, 1, 4 times

An alternative approach to the generation of spheres which can produce arbitrary number of neurons is to place neurons equispaced on a helix that sits on the surface of a sphere [3]. The identification and requirement of uniform neuron separations however becomes significantly more complicated than in the previous case. A third approach was shown by [6] where the Icosahedron method was used again but with a reduced time and space complexity for lookup within and creation of the neighborhood adjacency data structure.

2.1 Error Measures

Two distinct but complementary error measures are introduced here to quantify the quality of a SOM. Error measures for use in SOM are discussed more completely by [23] and [24].

The Quantisation Error (QE) represents the average difference between each pattern vector x_j and its best matching unit m_c and is calculated as follows.

$$QE = {}^{1}/_{n} \sum_{j=1}^{n} \|m_c - x_j\|$$

This measure does not however take into account the quality of the topological mapping for which a second error measure, Topological Error (TE) is used.

$$TE = {}^{1}/_{n} \sum_{j=1}^{n} \varphi(x_j, m_1, \cdots, m_p)$$

where $\varphi(x_j, m_1, \cdots, m_p) = \begin{cases} 1, \text{if 1st closest neighbor maps to 2nd closest} \\ 0, \quad otherwise \end{cases}$

3 Concentric Multi-Sphere SOMs and Connection Metrics

A key component of our approach is that the multiple spheres are concentric. This does not in general mean that inner spheres are smaller than outer spheres as would be the case for physical spheres. Rather, we consider that each neuron on a sphere has a nominal equivalence with "the same" neuron on each of its adjacent spheres. Notionally, each sphere in sequence is considered to be similar to next sphere which is at the next quantum of time distant. We can consider "space" and "time" to be topologically equivalent, which allows us to represent time via our concentric spheres. Our approach allows a number of possible connection metrics.

3.1 Self Metric

The 'Self' metric defines a sphere connection scheme where each neuron on each sphere is additionally connected to the equivalent neuron on some or all of the other spheres. Each inter-sphere connection is of length 1 regardless of which sphere maps to which. It will be seen that it is effective to conceptualize these spheres as sitting in a sequential order. Alternatively the arrangement could be thought of as multiple concentric spheres.

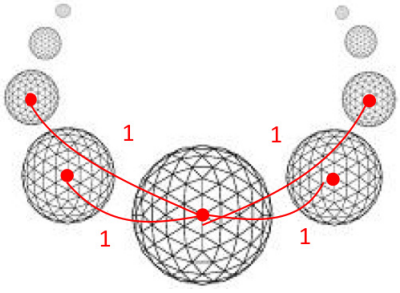


Fig. 2. Self Metric: 4 connected spheres with connection distances 1

In this metric the number of spheres connected is a parameter. For example in figure 2, four spheres are connected with two on each side. Alternatively this parameter could be specified as a fraction of the remaining spheres and rounded down to the nearest even number spheres. In this way a neuron can connect to as little as 2 spheres or potentially all spheres.

3.2 Distance Metric

On a single sphere it has often been decided to limit the maximum initial neighborhood size to half the sphere or one hemisphere [4]. The 'Distance' metric proposes that a virtual distance can be imagined to separate spheres. It should be noted that any neurons connected directly will always have a separation distance of 1 in terms of the SOM algorithm however the use of a virtual distance between spheres can be used to provide a more gradual tapering of the inter-sphere neighborhoods. This scheme is visualized is figure 3.

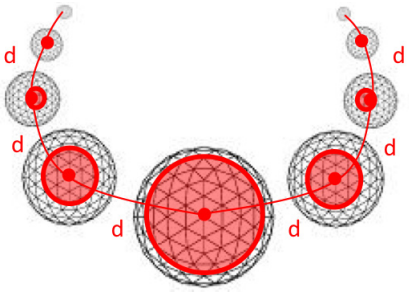


Fig. 3. Distance Metric: Connections of length *d*. Red circles indicate the neighborhoods of the center node.

The virtual distance d between spheres can be set such that the distance from a given node to the edge of the hemisphere should be the same as the distance travelled around half of the spheres. This is equivalent to $d = 2R/N_{sph}$. Here R is the distance (number of neurons) to the edge of the hemisphere and N_{sph} is the number of spheres. This metric is proposed to give a smooth decay in inter-sphere neighborhood connectivity.

3.3 Integer Metric

This is a simplified form of the distance metric which uses a constant value for *d*, and was our first notion for connecting spheres – with a 'time' distance of 1 equal to a 'space' distance of one. This metric can have a problem of 'wrapping around' if the spheres are in a cycle or not decaying fast enough if the spheres are merely linearly organised.

3.4 Data Order and Sphere Targeting

Our objective is to compare a number of distance metrics and error measures on our concentric multi-sphere SOMs (CMS-SOMs). With multiple spheres and a data set we can identify an number of plausible training regimes.

Data order refers to the presentation order of the input data set and can be randomised or preserved. For a time-series data set, we would expect better results if the data order is preserved if we learn useful time-based features in our model.

Sphere targeting refers to the restriction that the best matching neuron must be selected from a specific sphere. In this case during training the sphere targeted is incremented in order as the patterns are presented. The other option for targeting is 'any' where all spheres are candidates to select a winning neuron. For time-series data, we would expect better results from 'preserved, specific' if consistent time-based features were extracted. On the other hand, 'random, any' would provide little evidence for good learning of time-based features but would demonstrate the general power of our approach.

4 Data Set – ECSH

The Easy Calm Stressful Hard (ECSH) dataset contains eye gaze and physiological data collected while the subject is reading some paragraphs of text which are from these four categories. This dataset is from a larger project [11-12] on computational models for stress.

The ECSH dataset has 7 input fields being the x,y co-ordinates of the eye gaze, left and right pupil diameters, electrocardiogram (ECG), galvanic skin response (GSR) and instantaneous blood pressure (BP).

Biopac ECG100C, Biopac GSR100C and Finapres Finger Cuff systems were used to take ECG, GSR and blood pressure recordings at a sampling rate of 1000 Hz. Eye gaze and pupil dilation signals were obtained using a Seeing Machines FaceLAB system with a pair of infrared cameras at 60 Hz.

The data was synchronized and the ECSH dataset consists of 3,461 pattern vectors corresponding to a sampling rate of 60Hz. The data is classified into the four categories being the nature of the text being read, and is in 4 contiguous blocks of patterns.

5 Experiment and Results

In this experiment each inter-sphere connection metric is is used to train a range of numbers of spheres of different sizes and the Quantisation and Topological Errors are calculated for each combination of the data order and sphere targeting parameters.

Table 1. Shows the different cases of sphere numbers and sizes

No. spheres	Sphere structure	Neurons / sphere	Total neurons
1	$ICOSA_4$	2562	2562
4	$ICOSA_3$	642	2568
15	$ICOSA_2$	162	2430
61	$ICOSA_1$	42	2562
214	$ICOSA_0$	12	2568

Here $ICOSA_N$ refers to an icosahedron with N recursive subdivisions. We have chosen the structure and number of spheres so that the total number of neurons is as similar as possible in each case. Simulations were run for 50 epochs with the neighbourhood radius parameter R being 0.5. Error values are averaged over 10 runs.

5.1 Quantisation Error

The best result for QE on each sphere configuration in each metric is in bold. The best result for each sphere configuration (omitting self metrics) is also shaded in grey.

Table 2. QE – Integer metric

Data Order	Targeting	1 sphere	4 spheres	15 spheres	61 spheres	214 spheres
preserved	any	272.19	252.94	529.68	231.69	303.88
preserved	specific	563.82	292.80	624.78	1542.46	721.03
random	any	**176.57**	**168.70**	**283.12**	**126.86**	**166.95**
random	specific	204.12	183.40	721.81	3548.49	925.05

Random-any is clearly the best performing algorithm with lowest error for the 214 sphere case. (The best results for 61 spheres is random-any using self-metric with 4 connected spheres.) Random, specific performs quite poorly in general. In both cases with targeting of specific large errors are seen for the 61 sphere case..

Table 3. QE – Distance metric

Data Order	Targeting	1 sphere	4 spheres	15 spheres	61 spheres	214 spheres
preserved	any	188.18	274.19	**412.80**	**673.87**	626.99
preserved	specific	**132.94**	301.95	924.14	823.21	879.35
random	any	148.43	**157.40**	547.53	724.62	**625.38**
random	specific	179.99	245.15	895.46	8703.96	908.92

Here preserved, any and random any perform similarly indicating the data order is having little effect on learning. Disappointingly preserved, specific performs poorly again except in the case of a single sphere which is somewhat irrelevant as targeting is not very meaningful for a single sphere.

It is clear that Quantisation error has no way to measure time-based learning as it simply measures how closely the neurons map to the input vectors. It is proposed that random, any performs so well for the following two reasons. Firstly the random data presentation order may be beneficial to avoid any 'forgetting' problems if the data has some non-random structure. For example if the data is a constant value continually for most of the data set and only deviates in value for small subsection then randomly indexed may be beneficial to ensure the feature map trains on the different values at a more uniform rate. The second reason is that the specific targeting limits the feature

maps exposure to each data point as only a subset of the map is available for competition and adaption. This will probably reduce QE or at least slow its decrease over time whereas 'any' or not targeting facilitates a faster training and more complete exposure to the input data.

This hypothesis is also supported by the fact that there seems to be somewhat of a general decrease in QE as the amount of inter-sphere connectivity increases, i.e. as we transition from the Self metrics to the Distance metrics.

5.2 Topological Error

Here significantly low TE values are encountered for smaller number of spheres in all cases but performance degrades with an increasing number of spheres. Neither of the good results for preserved-specific on 1 sphere and the random-specific result are a clear signal of successful learning of time-based features.

Table 4. TE – Integer metric

Data Order	Targeting	1 sphere	4 spheres	15 spheres	61 spheres	214 spheres
preserved	any	0.078	0.049	0.237	**0.231**	**0.405**
preserved	specific	**0.067**	0.035	0.341	0.612	0.989
random	any	0.078	0.044	0.226	0.245	0.416
random	specific	0.078	**0.024**	**0.192**	0.691	0.991

Table 5. TE – Distance metric

Data Order	Targeting	1 sphere	4 spheres	15 spheres	61 spheres	214 spheres
preserved	any	**0.073**	0.210	0.228	0.158	0.098
preserved	specific	0.096	0.130	0.227	0.279	0.553
random	any	0.077	0.226	0.214	0.135	**0.096**
random	specific	0.094	**0.094**	**0.064**	**0.061**	0.564

TE seems to perform quite well over a large range of cases with little of the severe degradation in performance seen previously.

As with QE there seems to be some level of improved performance for greater inter-sphere connectivity. It is unclear whether TE is a suitable measure to quantify time learning due to the uncertainty whether the learning algorithms presented here are strongly capable of learning time-series characteristics. We will investigate further.

6 Conclusion and Future Work

We have used our concentric multi-sphere SOM structure to show that neither the Quantization error nor the Tological error measures are well suited to demonstrating

the quality of learning of time-based features on an SOM structure designed for that goal. We have introduced a scheme of DataOrder X Targeting to demonstrate this. Of the four cases, random-any clearly demonstrates the (non-time-based) power of our algorithm, preserved-specific would show good time-based feature learning well aligned to our SOM structure, preserved-any would show time-based feature learning less well aligned to our SOM structure (time-based features are learnt, but would imply some alternative topology could achieve better results), while finally random-specific is a general control and has no particular useful meaning, so a good result here implies both a better topology and distance metric / learning algorithm should be used.

Acknowledgements. The authors would like to express their appreciation to all the volunteers who participated in the data collection and user study.

References

1. Bação, F., Lobo, V.: Introduction to Kohonen's Self-Organizing Maps, Universidade Nova De Lisboa, Portugal (2010), http://edugi.uji.es/Bacao/SOM%20Tutorial.pdf
2. Kohonen, T.: Self-Organizing Maps. Springer (1995)
3. Nishio, H., Altaf-Ui-Amin, M.D., Kurokawa, K., Kanaya, S.: Spherical SOM and Arrangement of Neurons Using Helix on Sphere. IPSJ Digital Courier 2, 133–137 (2006)
4. Leontitsis, A., Sangole, A.P.: Estimating an optimal neighborhood size in the spherical self-organizing feature map. Int. Journal of Computational Intelligence 2, 94–98 (2005)
5. Sarle, W.S.: Neural Network FAQ Part 1 (2002), ftp://ftp.sas.com/pub/neural/FAQ.html
6. Wu, Y., Takatsuka, M.: Spherical self-organizing map using efficient indexed geodesic data structure. Neural Networks 19, 900–910 (2006)
7. Ritter, H.: Self-organizing maps on non-euclidean spaces. Kohonen Maps, 97–108 (1999)
8. Sangole, A., Knopf, G.K.: Visualization of randomly ordered numeric data sets using spherical self-organizing feature maps. Computers & Graphics 27, 963–976 (2003)
9. Sugii, Y., Satoh, H., Yu, D., Matsuura, Y., Tokutaka, H., Seno, M.: Spherical self-organizing map as a helpful tool to identify category-specific cell surface markers. Biochemical and Biophysical Research Communications 376 (2008)
10. Blazejewski, A., Coggins, R.: Application of self-organizing maps to clustering of high-frequency financial data. In: 2nd Workshop on Australasian Information Security, Data Mining and Web Intelligence, and Software Internationalisation, vol. 32, pp. 85–90 (2004)
11. Sharma, N., Gedeon, T.D.: Objective measures, sensors and computational techniques for stress recognition and classification: A survey. Computer Methods and Programs in Biomedicine 108(3), 1287–1301 (2012), doi:10.1016/j.cmpb.2012.07.003
12. Sharma, N., Gedeon, T.: Computational Models of Stress in Reading Using Physiological and Physical Sensor Data. In: Pei, J., Tseng, V.S., Cao, L., Motoda, H., Xu, G. (eds.) PAKDD 2013, Part I. LNCS (LNAI), vol. 7818, pp. 111–122. Springer, Heidelberg (2013)

Author Index